SIAM J. SCI. COMPUT.
Vol. 39, No. 2, pp. B424–B441
© 2017 Society for Industrial and Applied Mathematics

DYNAMIC EVALUATION OF FREE-FORM CURVES AND SURFACES*

XUNNIAN YANG† AND JIALIN HONG‡

Abstract. Free-form curves and surfaces represented by control points together with well-defined basis functions are widely used in computer aided geometric design. Efficient evaluation of points and derivatives of free-form curves and surfaces plays an important role in interactive rendering or CNC machining. In this paper we show that free-form curves with properly defined basis functions are the solutions of linear differential systems. By employing typical numerical methods for solving the differential systems, points and derivatives of free-form curves and surfaces can be computed in a dynamical way. Compared with traditional methods for evaluating free-form curves and surfaces there are two advantages of the proposed technique. First, the proposed method is universal and efficient for evaluating a large class of free-form curves and surfaces. Second, the evaluation needs only arithmetic operations even when the free-form curves and surfaces are defined using some transcendental functions.

Key words. free-form curves and surfaces, linear differential systems, evaluation, dynamic generation

AMS subject classifications. 65D17, 65D25, 65L05, 65L06

DOI. 10.1137/16M1058911

1. Introduction.

Free-form curves and surfaces constructed by blending control points with a set of basis functions have been widely used in the fields of geometric modeling, computer graphics, and computer aided manufacturing [6]. Besides polynomial based basis functions, the blending functions can also be transcendental functions. The nonpolynomial basis functions can be used to represent some typical curves without rational forms [37] or represent machining trajectories with low frequencies [9]. The basis functions other than polynomials used in the fields of geometric modeling and manufacturing include the exponential function [24], the trigonometric functions [3, 32, 29, 14], the hyperbolic functions [19], or their mixtures with polynomial functions [2, 34].

Efficient sampling and evaluation of points and derivatives of free-form curves and surfaces play important roles in fast rendering or high speed machining. Many methods have been developed for evaluating free-form curves and surfaces with polynomial bases. Points on polynomial or rational Bézier curves and surfaces can be evaluated robustly by the de Casteljau algorithm or the rational de Casteljau algorithm with simple arithmetic operations [6, 4]. Due to their fractal nature, Bézier curves can even be rendered by iterating from an arbitrary initial shape [10]. By converting the basis functions into power bases, Bézier or NURBS (Nonuniform rational B-spline) curves and surfaces can be rendered efficiently by employing forward differencing [18, 20, 8].

*Submitted to the journal's Computational Methods in Science and Engineering section February 8, 2016; accepted for publication (in revised form) December 21, 2016; published electronically April 25, 2017.
http://www.siam.org/journals/sisc/39-2/M105891.html
Funding: This work was supported by National Natural Science Foundation of China grants 11290142, 61272300. The second author was also supported by NSFC grants 91530118, 91630312, 91130003.
†Corresponding author. School of Mathematical Sciences, Zhejiang University, Hangzhou 310027, China (yxn@zju.edu.cn).
‡LSEC, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China (hjl@lsec.cc.ac.cn).

However, the differencing method cannot be extended to general types of free-form curves and surfaces. If the bases are defined by some transcendental functions the points and derivatives of the free-form curves and surfaces have to be evaluated by using specially designed procedures or devices [15, 25].

Several popular basis functions for constructing free-form curves and surfaces are chosen as the elements of null spaces of certain constant coefficient differential operators [12, 16]. Based on the nice properties of the differential operators, blossoming or recursive evaluation algorithms can be derived for the basis functions. In this paper we show that free-form curves represented by control points and a large class of basis functions are the solutions of *linear differential systems* when the spaces spanned by the basis functions are closed with respect to a differential operation. By employing typical numerical methods such as the Taylor method or the implicit midpoint scheme to solve the differential systems, the points and derivatives of free-form curves can be calculated exactly or approximately with high accuracies. It is also shown that a tensor-product surface can be generated by evaluating a set of isoparametric curves through solving linear differential systems. By representing free-form curves as the solutions of linear differential systems, free-form curves defined in various function spaces can be evaluated in the same way. Since the linear differential systems can be solved using only elementary arithmetic operations, free-form curves and surfaces defined by polynomials as well as transcendental basis functions can be evaluated or rendered on general computing machines.

The rest of this paper is structured as follows. In section 2 we show that a wide class of free-form curves used in computer aided geometric design are the solution curves of linear differential systems. A dynamic approach to evaluating free-form curves and surfaces is introduced in section 3 by employing popular numerical methods to solve the differential systems. Several interesting examples and comparisons with some existing methods of curve generation are given in section 4. Section 5 concludes the paper with a brief summary of our work.

2. Representing free-form curves as the solutions of linear differential systems. In this section we show that free-form curves that are constructed by blending control points with basis functions are the solutions of linear differential systems when the spaces spanned by the basis functions are closed with respect to a differential operation. Rational curves are also the solutions of differential systems when the points are represented by homogeneous coordinates.

2.1. Construction of linear differential systems for representing free-form curves in $\mathbb{R}^{n+1}$. A control point based free-form curve in $\mathbb{R}^d$ can be represented by

$$X(t) = \sum_{i=0}^{n} X_i \phi_i(t), \quad t \in [\alpha, \beta], \tag{1}$$

where $X_i \in \mathbb{R}^d$, $i = 0, 1, \ldots, n$, are the control points and $\phi_i(t)$, $i = 0, 1, \ldots, n$, are the blending or the basis functions. In particular, if the functions $\phi_i(t)$ in (1) are chosen as the Bernstein basis functions, i.e., $\phi_i(t) = B_{i,n}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$, $t \in [0, 1]$, $i = 0, 1, \ldots, n$, the free-form curve $X(t)$ is just the well-known Bézier curve [7]. To represent circles, cylinders, or more general types of curves and surfaces with periodic property in nonrational form, the blending functions are constructed by trigonometric functions [30] or a combination of trigonometric functions and polynomials [37, 3, 21]. If hyperbolic curves are generated, the blending functions include the hyperbolic

Table 1
Free-form curves and linear spaces spanned by their basis functions.

Free-form curves	The space of basis functions (Ω)
Bézier curves [1, 7]	$\text{span}\{B_{0,n}(t), B_{1,n}(t), \ldots, B_{n,n}(t)\}$
C-curves [37]	$\text{span}\{1, t, \cos t, \sin t\}$
Trigonometric curves [31]	$\text{span}\{1, \cos t, \sin t, \ldots, \cos nt, \sin nt\}$
Hyperbolic curves [33]	$\text{span}\{1, \cosh t, \sinh t, \ldots, \cosh nt, \sinh nt\}$
Involute compatible [22]	$\text{span}\{1, t, \cos t, \sin t, t\cos t, t\sin t\}$
Algebraic-trigonometric [3]	$\text{span}\{1, t, \ldots, t^{n-2}, \cos t, \sin t\}$
Algebraic-hyperbolic [17]	$\text{span}\{1, t, \ldots, t^{n-2}, \cosh t, \sinh t\}$
Hyperbolic-trigonometric [2]	$\text{span}\{1, \cosh t, \sinh t, \cos t, \sin t\}$
AHT Bézier curves [36]	$\text{span}\{1, t, \ldots, t^{n-5}, \cosh t, \sinh t, \cos t, \sin t\}$
Intrinsically defined [35]	$\text{span}\{1, \cos t, \sin t, t\cos t, t\sin t, \ldots, t^n\cos t, t^n\sin t\}$

functions [17, 36, 2]. Several typical blending functions constructed by polynomials, transcendental functions, or their mixtures are summarized in Table 1.

Using linear combinations of the initial algebraic or transcendental functions, a system of normalized totally positive (NTP) bases can be constructed for a known function space. A change of basis does not change the vector space spanned by the original bases and the free-form curves defined by NTP bases follow the shapes of their control polygons very well [26, 28]. Besides the description of free-form curves using control polygons, a fair planar curve can be represented using ordinary bases when it is designed based on an intrinsic expression of planar curves [35]. From Table 1 we can easily check that the space Ω spanned by each set of basis functions is closed with respect to differentiation by d/dt. We show that the free-form curves constructed by these basis functions are the solutions of differential systems, and then new methods will be developed for evaluating the free-form curves and surfaces based on the new representation.

Theorem 1. *Suppose that $\phi_i(t)$, $i = 0, 1, \ldots, n$, are a set of basis functions for a linear space Ω and $\phi_i'(t) \in \Omega$ for $i = 0, 1, \ldots, n$. Let $X_i = (x_{0i}, x_{1i}, \ldots, x_{ni})^T$, $i = 0, 1, \ldots, n$, be a set of points in $\mathbb{R}^{n+1}$. If the matrix $M_X = (X_0, X_1, \ldots, X_n)$ is nonsingular, the curve $X(t) = \sum_{i=0}^n X_i\phi_i(t)$, $t \in [\alpha, \beta]$, is one solution curve of a linear differential system.*

Proof. To prove the theorem we first rewrite the free-form curve as

$$X(t) = (X_0, X_1, \ldots, X_n)\begin{pmatrix} \phi_0(t) \\ \phi_1(t) \\ \vdots \\ \phi_n(t) \end{pmatrix}.$$

Under the assumption that the matrix $M_X = (X_0, X_1, \ldots, X_n)$ is nonsingular, the basis vector can be reformulated as

$$\begin{pmatrix} \phi_0(t) \\ \phi_1(t) \\ \vdots \\ \phi_n(t) \end{pmatrix} = (X_0, X_1, \ldots, X_n)^{-1}X(t).$$

On the other hand, because $\phi_i'(t) \in \Omega$ we have

$$\phi_i'(t) = (c_{i0}, c_{i1}, \ldots, c_{in}) \begin{pmatrix} \phi_0(t) \\ \phi_1(t) \\ \vdots \\ \phi_n(t) \end{pmatrix}$$

for $i = 0, 1, \ldots, n$. Hence, the derivative of $X(t)$ is

$$\begin{aligned} X'(t) &= (X_0, X_1, \ldots, X_n) \begin{pmatrix} \phi_0'(t) \\ \phi_1'(t) \\ \vdots \\ \phi_n'(t) \end{pmatrix} \\ &= (X_0, X_1, \ldots, X_n) \begin{pmatrix} c_{00} & c_{01} & \cdots & c_{0n} \\ c_{10} & c_{11} & \cdots & c_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n0} & c_{n1} & \cdots & c_{nn} \end{pmatrix} \begin{pmatrix} \phi_0(t) \\ \phi_1(t) \\ \vdots \\ \phi_n(t) \end{pmatrix}. \end{aligned}$$

By substituting $(X_0, X_1, \ldots, X_n)^{-1}X(t)$ for the basis vector, we obtain a linear differential system

$$X'(t) = AX(t),$$

where

$$A = (X_0, X_1, \ldots, X_n) \begin{pmatrix} c_{00} & c_{01} & \cdots & c_{0n} \\ c_{10} & c_{11} & \cdots & c_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n0} & c_{n1} & \cdots & c_{nn} \end{pmatrix} (X_0, X_1, \ldots, X_n)^{-1}. \tag{2}$$

This proves the theorem. □

From Theorem 1 we know that a free-form curve $X(t) = \sum_{i=0}^{n} X_i \phi_i(t)$ passing through a point $X_{orig} \in \mathbb{R}^{n+1}$ at $t = \alpha$ is just the solution curve of the differential system

$$\begin{cases} X'(t) = AX(t), & t \in [\alpha, \beta], \\ X(\alpha) = X_{orig}, \end{cases} \tag{3}$$

where the coefficient matrix A is given by (2).

2.2. Construction of linear differential systems for representing free-form curves in $\mathbb{R}^2$ or $\mathbb{R}^3$. For practical modeling free-form curves are often defined on a plane or in three-dimensional (3D) space. Except for a few special cases, the number of control points of a planar or a spatial free-form curve is usually greater than the dimension of the space in which the curve is defined. To construct a differential system for representing these free-form curves, we take a similar technique as in [10], lifting the control points and the curves as well to a space of higher dimension. When a lifted curve has been evaluated, the points of the original curve are computed by orthogonal projection of the sampled points onto the plane or the 3D space.

We illustrate the method by constructing differential systems for representing free-form curves in $\mathbb{R}^3$. Differential systems for representing planar curves can be constructed in a similar way.

Suppose that $P_i = (x_i, y_i, z_i)^T$, $i = 0, 1, \dots, n$, are a sequence of control points in 3D space and a free-form curve is defined by

$$P(t) = \sum_{i=0}^{n} P_i \phi_i(t), \quad t \in [\alpha, \beta].$$

It is known that the matrix $M_P = (P_0, P_1, \dots, P_n)$ is not a square matrix for $n \geq 3$ and the coefficient matrix of the differential system (3) can no longer be derived by (2). Nevertheless, the free-form curve can still be represented as the solution curve of a differential system by lifting the control points to $\mathbb{R}^{n+1}$ when the original matrix M_P has a rank of 3.

Without loss of generality we assume that the 3×3 matrix composed of points P_{n-2}, P_{n-1}, and P_n has full rank. We lift the control points from $\mathbb{R}^3$ to $\mathbb{R}^{n+1}$ by choosing $X_0 = (x_0, y_0, z_0, 1, 0, \dots, 0)^T$, $X_1 = (x_1, y_1, z_1, 0, 1, \dots, 0)^T$, $\dots$, $X_{n-3} = (x_{n-3}, y_{n-3}, z_{n-3}, 0, 0, \dots, 1)^T$, and $X_i = (x_i, y_i, z_i, 0, 0, \dots, 0)^T$ for $i = n-2, n-1, n$. Thus, $X(t) = \sum_{i=0}^{n} X_i \phi_i(t)$ is a curve in $\mathbb{R}^{n+1}$. Let

$$M_X = \begin{pmatrix} x_0 & x_1 & \cdots & x_{n-3} & x_{n-2} & x_{n-1} & x_n \\ y_0 & y_1 & \cdots & y_{n-3} & y_{n-2} & y_{n-1} & y_n \\ z_0 & z_1 & \cdots & z_{n-3} & z_{n-2} & z_{n-1} & z_n \\ 1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & 0 \end{pmatrix}. \tag{4}$$

Since the matrix (P_{n-2}, P_{n-1}, P_n) is nonsingular, the matrix M_X is also regular. Denote the matrix M_X as a block matrix

$$\begin{pmatrix} P_I & P_{II} \\ I_{n-2} & 0 \end{pmatrix},$$

where I_{n-2} is the identity matrix of order $n-2$, P_I and P_{II} are the corresponding submatrices of M_X. The inverse of the matrix M_X is

$$M_X^{-1} = \begin{pmatrix} 0 & I_{n-2} \\ P_{II}^{-1} & -P_{II}^{-1} P_I \end{pmatrix}.$$

According to Theorem 1, $X(t)$ is the solution curve of a differential system as given by (2) and (3) where the matrix $(X_0, X_1, \dots, X_n)$ is now defined by (4). Since the solution curve of the differential system lies in $\mathbb{R}^{n+1}$, the original curve $P(t)$ can be restored by choosing the first three coordinates of $X(t)$.

From a geometric point of view, if a line that passes through two distinct points on the xy-plane does not pass through the origin, the 2×2 matrix composed of the coordinates of the two points is not singular. Similarly, if three points in 3D space do not lie on the same line and the plane spanned by the three points does not pass through the origin, the 3×3 matrix composed of the coordinates of the three points has full rank. For a free-form curve $P(t)$ defined on the xy-plane or in 3D space, we have the following proposition.

PROPOSITION 2. *Suppose that the linear space* $\Omega = \operatorname{span}\{\phi_0(t), \phi_1(t), \dots, \phi_n(t)\}$ *is closed with respect to a differential operation and* $P(t) = \sum_{i=0}^{n} P_i \phi_i(t)$ *is either a* *curve lying on the xy-plane or a curve in 3D space. If a line that passes through two distinct control points on the xy-plane or a plane spanned by three control points in the 3D space does not pass through the origin, the lifted curve in $\mathbb{R}^{n+1}$ is the solution curve of a differential system.*

Besides orthogonal projection, a free-form curve in $\mathbb{R}^2$ or $\mathbb{R}^3$ can also be generated by a projective transformation of a normal curve which lies in a space of higher dimension [11]. Let E_i, $i = 0, 1, \ldots, n$, be the ith column vector of an identity matrix of order $n + 1$; a normal curve in $\mathbb{R}^{n+1}$ is given by $\varphi(t) = \sum_{i=0}^{n} E_i\phi_i(t) = (\phi_0(t), \phi_1(t), \ldots, \phi_n(t))^T$. The free-form curve $P(t) = \sum_{i=0}^{n} P_i\phi_i(t)$, $t \in [\alpha, \beta]$, is then generated by the projective transformation $P(t) = (P_0, P_1, \ldots, P_n)\varphi(t)$. Based on Theorem 1, $\varphi(t)$ is the solution curve of a linear differential system. When $\varphi(t)$ and its derivatives are evaluated by solving a differential system, points and derivatives on $P(t)$ will be generated by the projective transformation. Even though it is guaranteed that a differential system can be constructed to represent a normal curve, the orthogonal projection method is still preferred for evaluating free-form curves since there is no need to compute a projective transformation.

2.3. Construction of linear differential systems for representing rational curves. Besides their nonrational counterparts, rational curves and surfaces are still mandatory for shape representation and geometric modeling [5]. Some typical curves can even be represented exactly by using rational transcendental functions [14]. Suppose that the real weights ω_i, $i = 0, 1, \ldots, n$, of rank 1 are associated with the control points $P_i \in \mathbb{R}^d$, $i = 0, 1, \ldots, n$. A rational curve in $\mathbb{R}^d$ is defined by

$$R(t) = \frac{\sum_{i=0}^{n} \omega_i P_i \phi_i(t)}{\sum_{i=0}^{n} \omega_i \phi_i(t)}, \quad t \in [\alpha, \beta], \tag{5}$$

where $\phi_i(t)$ are basis functions that are defined by polynomials or by a mixture of polynomials and transcendental functions. If the denominator does not vanish for any $t \in [\alpha, \beta]$ the rational curve $R(t)$ is *valid* over the parameter domain.

A rational curve can also be defined by control points together with a set of rational basis functions, i.e., $R(t) = \sum_{i=0}^{n} P_i\bar{\phi}_i(t)$, where $\bar{\phi}_i(t) = \frac{\omega_i\phi_i(t)}{\sum_{j=0}^{n}\omega_j\phi_j(t)}$, $i = 0, 1, \ldots, n$. Unless all denominators of functions $\bar{\phi}_i(t)$ are equal to a constant, the space spanned by the rational basis functions is not closed with respect to a differential operation. Then, the rational curve $R(t)$ is not the solution of any explicit differential system. Nevertheless, a differential system can still be constructed for representing the original rational curve by employing homogeneous coordinates. Let $\widetilde{P}_i = (\omega_i P_i^T, \omega_i)^T$, $i = 0, 1, \ldots, n$; a nonrational curve in $\mathbb{R}^{d+1}$ is defined as $\widetilde{R}(t) = \sum_{i=0}^{n} \widetilde{P}_i\phi_i(t)$. Thus, the rational curve $R(t)$ is the projection of $\widetilde{R}(t)$ onto hyperplane $\omega = 1$ [6]. When points on the curve $\widetilde{R}(t)$ have been evaluated, the points on the rational curve $R(t)$ can be generated by projecting the points from $\mathbb{R}^{d+1}$ onto hyperplane $\omega = 1$.

Since a rational curve with two or three control points is just a line or a planar curve which can be generated by even simpler methods, we consider the construction of differential systems for representing planar rational curves that have at least three control points or representing spatial rational curves that have at least four control points.

PROPOSITION 3. *Suppose that the linear space* $\Omega = \mathrm{span}\{\phi_0(t), \phi_1(t), \ldots, \phi_n(t)\}$ *is closed with respect to a differential operation and* $R(t) = \frac{\sum_{i=0}^{n}\omega_i P_i\phi_i(t)}{\sum_{i=0}^{n}\omega_i\phi_i(t)}$, $t \in [\alpha, \beta]$, *is a valid rational curve lying on the xy-plane or in 3D space, where* $\omega_i \neq 0$ *are the* *weights and $P_i = (x_i, y_i)^T$ or $P_i = (x_i, y_i, z_i)^T$ are the control points. If all of the control points do not lie on the same line on the xy-plane or all control points do not lie on the same plane in the 3D space, the preimage of the rational curve $R(t)$ is the solution curve of a differential system.*

Proof. We prove the proposition for the planar case; the differential systems for representing rational curves in 3D space can be constructed in a similar way.

Suppose that a rational curve $R(t)$ lies on the xy-plane; the homogeneous coordinates of the control points are $(\omega_i x_i, \omega_i y_i, \omega_i)^T$, $i = 0, 1, \dots, n$. Assume that $n \geq 2$ and the last three control points do not lie on the same line on the xy-plane, which implies that the vectors $P_n - P_{n-2}$ and $P_{n-1} - P_{n-2}$ are not parallel. Then the determinant

$$\begin{vmatrix} \omega_{n-2}x_{n-2} & \omega_{n-1}x_{n-1} & \omega_n x_n \\ \omega_{n-2}y_{n-2} & \omega_{n-1}y_{n-1} & \omega_n y_n \\ \omega_{n-2} & \omega_{n-1} & \omega_n \end{vmatrix} = \omega_{n-2}\omega_{n-1}\omega_n \begin{vmatrix} x_{n-1} - x_{n-2} & x_n - x_{n-2} \\ y_{n-1} - y_{n-2} & y_n - y_{n-2} \end{vmatrix} \neq 0.$$

By lifting the points $(\omega_i x_i, \omega_i y_i, \omega_i)^T$ to a space of dimension $n+1$, the matrix composed of the lifted control points is

$$\begin{pmatrix} \omega_0 x_0 & \omega_1 x_1 & \cdots & \omega_{n-3}x_{n-3} & \omega_{n-2}x_{n-2} & \omega_{n-1}x_{n-1} & \omega_n x_n \\ \omega_0 y_0 & \omega_1 y_1 & \cdots & \omega_{n-3}y_{n-3} & \omega_{n-2}y_{n-2} & \omega_{n-1}y_{n-1} & \omega_n y_n \\ \omega_0 & \omega_1 & \cdots & \omega_{n-3} & \omega_{n-2} & \omega_{n-1} & \omega_n \\ 1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Denote the column vectors of the above matrix from left to right as X_i, $i = 0, 1, \dots, n$. The lifted free-form curve in $\mathbb{R}^{n+1}$ is $X(t) = \sum_{i=0}^{n} X_i \phi_i(t)$. Since the matrix composed of the lifted control points is nonsingular, a differential system for which the lifted free-form curve $X(t)$ is a solution can be constructed by the same technique as in section 2.2. ∎

Popular rational spline curves such as NURBS curves can always be decomposed into piecewise rational curves using the knot insertion technique [27]. After knot insertion, each piece of a rational curve is represented as the solution curve of an individual differential system.

3. Evaluating free-form curves and surfaces by solving linear differential systems. In this section we pay attention to techniques for generating free-form curves by numerically solving linear differential systems. Free-form surface generation by computing points and derivatives of series of free-form curves will also be discussed.

3.1. Dynamic evaluation of free-form curves. The solution of differential system (3) can be formulated as $X(t) = e^{A(t-\alpha)}X_{orig}$. However, the solution curve can be evaluated explicitly only when the exponential of the coefficient matrix has an explicit expression. Though many methods have been given in the literature to tackle this problem, the exponentials of matrices are not always computed by simple elementary functions [23]. Even if the coefficient matrix A is diagonalizable, the elements of the exponential matrix are still represented by transcendental functions which eventually need special procedures to compute. Instead of computing exponentials of matrices, in this paper we evaluate free-form curves and surfaces by employing efficient numerical methods to solve the linear differential systems. The numerical methods need only arithmetic operations and can give high accuracy or even exact results for general types of linear differential systems.

The first method we use to solve the differential system (3) is the Taylor method. This method can give exact solutions when the solution curves of the differential system are polynomial curves and the technique usually gives high-accuracy results for other types of solution curves. Assume K points are evaluated on the interval $[\alpha, \beta]$ except for the initial point X_{orig}; the time step can be chosen as $\Delta t = \frac{\beta - \alpha}{K}$. Starting from the initial point $Y_0 = X_{orig}$, all other points Y_i, $i = 1, 2, \ldots, K$, are computed sequentially.

Suppose that $X(t_i)$ is a point on the curve and let $h = \Delta t$. The point $X(t_i + h)$ is computed by the Taylor method as

$$\begin{aligned} X(t_i + h) &= X(t_i) + hX'(t_i) + \frac{h^2}{2!}X''(t_i) + \cdots + \frac{h^s}{s!}X^{(s)}(t_i) + o(h^s) \\ &= X(t_i) + hAX(t_i) + \frac{h^2A^2}{2!}X(t_i) + \cdots + \frac{h^sA^s}{s!}X(t_i) + o(h^s) \\ &= M_hX(t_i) + o(h^s), \end{aligned}$$

where $M_h = I + hA + \frac{h^2A^2}{2!} + \cdots + \frac{h^sA^s}{s!}$ and I is the identity matrix of order $n+1$. Since the coefficient matrix M_h is independent of any specially chosen point, the numerical solution of (3) is

$$Y_i = M_hY_{i-1}, \quad i = 1, 2, \ldots, K. \tag{6}$$

From the above derivation we know that if $X(t)$ is a polynomial curve of degree n, the discrete points generated by (6) lie on the original curve exactly when $s \geq n$.

Another simple but efficient method we can use to solve the differential system (3) is the implicit midpoint scheme [13]. Suppose that $X(t_i)$ and $X(t_i + \Delta t)$ are two neighboring points on the curve. The differential system (3) is discretized as

$$\frac{X(t_i + \Delta t) - X(t_i)}{\Delta t} = A\frac{X(t_i + \Delta t) + X(t_i)}{2}.$$

If $X(t_i)$ is known, $X(t_i + \Delta t)$ can then be estimated by solving this discrete system. We generalize the implicit midpoint scheme for computing the discrete solutions of the differential system (3). Let $\Delta t = \frac{\beta - \alpha}{K}$ be the time step and $Y_i, i = 0, 1, \ldots, K$, be the sequence of points to be estimated with time step Δt. Starting from the initial point $Y_0 = X_{orig}$, the remaining points are computed by solving the equation

$$\frac{Y_i - Y_{i-1}}{h(\Delta t)} = A\frac{Y_i + Y_{i-1}}{2}, \tag{7}$$

where $h(\Delta t)$ is a function that satisfies $h(0) = 0$ and $h'(0) > 0$. By solving for Y_i from (7) we obtain the iteration formula

$$Y_i = \left(I - \frac{h(\Delta t)}{2}A\right)^{-1}\left(I + \frac{h(\Delta t)}{2}A\right)Y_{i-1}, \quad i = 1, 2, \ldots, K, \tag{8}$$

for the discrete solutions. Equation (8) reduces to the implicit midpoint scheme by choosing $h(\Delta t) = \Delta t$. Next we show that points of a quadratic curve at selected parameters can be evaluated exactly by solving a differential system using the iteration formula (8).

PROPOSITION 4. *Suppose that* $X_i = (x_{0i}, x_{1i}, x_{2i})^T$, $i = 0, 1, 2$, *are points that satisfy* $\det(X_0, X_1, X_2) \neq 0$ *and a quadratic curve is defined by* $X(t) = \sum_{i=0}^{2} X_i B_{i,2}(t)$, *where* $B_{i,2}(t)$ *are the Bernstein basis functions. By choosing* $h(\Delta t) = \Delta t$, *the points sampled on the curve with time step* Δt *can be computed sequentially by solving a linear differential system using* (8).

Proof. Similar to the derivation of (2) or (9), the quadratic curve $X(t) = \sum_{i=0}^{2} X_i B_{i,2}(t)$ is just the solution curve of the differential system

$$\begin{cases} X'(t) = AX(t), & t \in [0, 1], \\ X(0) = X_0, \end{cases}$$

where

$$A = (X_0, X_1, X_2) \begin{pmatrix} -2 & -1 & 0 \\ 2 & 0 & -2 \\ 0 & 1 & 2 \end{pmatrix} (X_0, X_1, X_2)^{-1}.$$

Let t_1 and t_2 be two distinct parameters in $[0, 1]$ and consider the sample points $X(t_1)$ and $X(t_2)$. To prove that the sample points satisfy (7) or (8), we show that

$$\frac{X(t_2) - X(t_1)}{t_2 - t_1} = A\frac{X(t_2) + X(t_1)}{2}.$$

By substituting $X(t_1) = \sum_{i=0}^{2} X_i B_{i,2}(t_1)$ and $X(t_2) = \sum_{i=0}^{2} X_i B_{i,2}(t_2)$ into the above equation, both sides of the equation are equal to $(X_0, X_1, X_2)(-(2 - t_1 - t_2), 2(1-t_1-t_2), t_1+t_2)^T$. This proves that the point $X(t+\Delta t)$ can be computed exactly by (8) from a known point $X(t)$. ∎

Besides evaluating quadratic curves, the generalized midpoint scheme can also be used to exactly compute points either on circles or on spheres.

PROPOSITION 5. *Suppose that the coefficient matrix* A *of* (3) *is a* 3×3 *skew-symmetric matrix. Then the solution curve* $X(t)$ *of the differential system is a circle. If* $h(\Delta t) = 2\tan(\frac{\Delta t}{2})$, *the points computed by* (8) *are* $Y_i = X(\alpha+i\Delta t)$, $i = 0, 1, \ldots, K$.

Proof. Without loss of generality, we assume that the skew-symmetric matrix is

$$A = \begin{pmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{pmatrix},$$

where n_x, n_y, and n_z are the components of a unit vector $\mathbf{n}$ in 3D space. Based on Rodrigues' rotation formula[1] the differential system (3) can be reformulated as $X'(t) = AX(t) = \mathbf{n}\times X(t)$. This differential equation implies that the derivative of the curve $X(t)$ is perpendicular to the vector $\mathbf{n}$ and the curve $X(t) = X(\alpha) + \int_{\alpha}^{t} X'(\xi)d\xi$ lies on the plane that passes through the point $X(\alpha)$ with normal $\mathbf{n}$. On the other hand, $X(t) \cdot X'(t)$ can be computed by $X^T(t)X'(t) = X^T(t)AX(t)$ or $[X'(t)]^T X(t) = X^T(t)A^T X(t) = -X^T(t)AX(t)$. Therefore, we have $X(t) \cdot X'(t) = \frac{1}{2}[X^2(t)]' = 0$, which gives $X^2(t) \equiv X_{orig}^2$. Since $X(t)$ lies on a plane and a sphere simultaneously, $X(t)$ is a circle. Suppose that the radius of the circle is r, the Euclidean norm of $X(t)$ is R, and the angle between $\mathbf{n}$ and $X(t)$ is θ. Then $r = R\sin\theta$.

[1]http://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula.

Replacing $X(t)$ by $X'(t)$ within $X(t) \cdot X'(t) = 0$, we have $X'(t) \cdot X''(t) = 0$ and $[X'(t)]^2 = \|X'(t)\|^2 \equiv$ const. This shows that the curve $X(t)$ has uniform parametrization and t is proportional to the arc length or the central angle of the circle. Since the length of the derivative $X'(t)$ is computed by $\|X'(t)\| = \|\mathbf{n} \times X(t)\| = R \sin\theta = r$, the parameter t is just the central angle of the circle. Assuming $X(t)$ and $X(t+\Delta t)$ are two distinct points on the circle, the unit vector corresponding to the chord from $X(t)$ to $X(t+\Delta t)$ is $\frac{X(t+\Delta t)-X(t)}{2r\sin\frac{\Delta t}{2}}$. Because $\|X'(t)\| = \|X'(t+\Delta t)\| = r$ and the angle between $X'(t)$ and $X'(t+\Delta t)$ is Δt, the unit vector corresponding to the average of tangents $X'(t)$ and $X'(t+\Delta t)$ is $\frac{X'(t)+X'(t+\Delta t)}{2r\cos\frac{\Delta t}{2}}$. Since the two unit vectors are the same, we have

$$\frac{X(t+\Delta t)-X(t)}{2r\sin\frac{\Delta t}{2}} = \frac{X'(t)+X'(t+\Delta t)}{2r\cos\frac{\Delta t}{2}}.$$

This equation can be simplified to

$$\frac{X(t+\Delta t)-X(t)}{2\tan\frac{\Delta t}{2}} = A\frac{X(t)+X(t+\Delta t)}{2}.$$

By choosing $h(\Delta t) = 2\tan\frac{\Delta t}{2}$, the points computed by (8) are just the points sampled on the curve $X(t)$ with fixed time step Δt. ∎

From Proposition 5 if we know the generalized time step $h(\Delta t)$, the points on a circle can be computed sequentially just by elementary arithmetic operations. In particular, $h(\Delta t) = 2\tan\frac{\Delta t}{2}$ can be evaluated to within any high precision by Taylor expansion, which needs only arithmetic operations.

When a sequence of points have been evaluated by either of the mentioned numerical methods, the derivatives of the curve at the evaluated points can be computed immediately based on the differential system. In particular, $A^l Y_i$, $l = 1, 2, \ldots$, are the lth derivatives of the curve at point Y_i.

The time complexity for evaluating a free-form curve by solving a linear differential system using either the Taylor method or the implicit midpoint scheme can be analyzed as follows. Assuming the coefficient matrix A of (3) is of order $n+1$, the coefficient matrices of (6) or (8) are of order $n+1$, too. If the time step for solving a linear differential system is fixed, the coefficient matrices of (6) or (8) are unchanged when a sequence of points are evaluated. By computing the coefficient matrices in advance, evaluation of each point Y_i needs $(n+1)^2$ multiplications and $n(n+1)$ additions by either of the two numerical schemes.

3.2. Dynamic generation of free-form surfaces. Based on the techniques of dynamic generation of free-form curves, a tensor-product surface can be generated by evaluating a series of isoparametric curves or a grid of points.

Suppose $C_i(v)$, $i = 0, 1, \ldots, n$, are a sequence of curves defined by

$$C_i(v) = \sum_{j=0}^{m} X_{ij} G_j(v),$$

where X_{ij} are the control points and $G_j(v)$, $c \le v \le d$, are the blending or the basis functions. Assuming $F_i(u)$, $a \le u \le b$, $i = 0, 1, \ldots, n$, are another set of blending or basis functions, a tensor-product surface can then be defined by

$$\begin{aligned} X(u,v) &= \textstyle\sum_{i=0}^{n} C_i(v) F_i(u) \\ &= \textstyle\sum_{i=0}^{n} \sum_{j=0}^{m} X_{ij} F_i(u) G_j(v). \end{aligned}$$

Before generating the free-form surface $X(u,v)$, we first evaluate all curves $C_i(v)$ with the same parameter step. Let $\Delta v = \frac{d-c}{L}$ and $v_l = c + l\Delta v$. The points $C_i(v_l)$, $0 \le l \le L$, can be evaluated for every curve $C_i(v)$ by employing the curve generation techniques stated in section 3.1. When the points $C_i(v_l)$, $i = 0, 1, \dots, n$, $l = 0, 1, \dots, L$, have been computed, isoparametric curves $X(u, v_l) = \sum_{i=0}^{n} C_i(v_l) F_i(u)$, $0 \le l \le L$, are then defined explicitly. These isoparametric curves can be evaluated numerically by employing the differential system representation.

The partial derivative $\frac{\partial X(u,v_l)}{\partial u}$ can be evaluated based just on the differential representation of the isoparametric curve and the evaluated point of $X(u, v_l)$. If the derivative $\frac{\partial X(u,v_l)}{\partial v}$ should be evaluated at the selected points, $C_i'(v_l)$ will first be computed along with the evaluation of $C_i(v_l)$. The derivative $\frac{\partial X(u,v_l)}{\partial v}$ can then be computed by evaluating the curve $\sum_{i=0}^{n} C_i'(v_l) F_i(u)$.

4. Examples and comparisons. The proposed algorithm for evaluating free-form curves and surfaces was implemented using C++ on a laptop with an Intel Core i7-4910MQ CPU@2.90 GHz 2.89 GHz and 8G RAM. Comparisons between the proposed method and some existing algorithms will be given.

Example 1. The first example is about dynamic evaluation of Bézier curves. Suppose that $X(t) = \sum_{i=0}^{n} X_i B_{i,n}(t)$ is a Bézier curve in $\mathbb{R}^{n+1}$. The derivative of the curve can be computed by $X'(t) = \sum_{i=0}^{n} X_i B_{i,n}'(t)$. Furthermore, the derivative of the basis vector can be formulated as

$$\begin{pmatrix} B_{0,n}'(t) \\ B_{1,n}'(t) \\ B_{2,n}'(t) \\ \vdots \\ B_{n-1,n}'(t) \\ B_{n,n}'(t) \end{pmatrix} = \begin{pmatrix} -n & -1 & 0 & \cdots & 0 & 0 \\ n & -n+2 & -2 & \cdots & 0 & 0 \\ 0 & n-1 & -n+4 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & n-2 & -n \\ 0 & 0 & 0 & \cdots & 1 & n \end{pmatrix} \begin{pmatrix} B_{0,n}(t) \\ B_{1,n}(t) \\ B_{2,n}(t) \\ \vdots \\ B_{n-1,n}(t) \\ B_{n,n}(t) \end{pmatrix}.$$

Denote the matrix before the basis vector as C_n. Let $M_X = (X_0, X_1, \dots, X_n)$ and $A = M_X C_n M_X^{-1}$. The Bézier curve is the solution of the linear differential system

$$\left\{ \begin{array}{lcl} X'(t) & = & AX(t), \quad t \in [0,1], \\ X(0) & = & X_0. \end{array} \right. \tag{9}$$

For a planar Bézier curve of degree at least 2 or a spatial Bézier curve of degree at least 3 the control points should be lifted to $\mathbb{R}^{n+1}$ using the technique discussed in section 2.2.

From section 3.1 we know that evaluating a point on a Bézier curve of degree n with a fixed time step needs $(n+1)^2$ multiplications and $n(n+1)$ additions by the proposed dynamic approach. Evaluating a point on a Bézier curve of degree n in $\mathbb{R}^d$ by the traditional de Casteljau algorithm needs $n(n+1)d$ multiplications and $n(n+1)d/2$ additions. Therefore, evaluating points on Bézier curves of degree greater than or equal to 2 the dynamic approach usually needs much less time than the traditional de Casteljau algorithm. Table 2 illustrates the time costs for generating Bézier curves of degree 3, 5, 8, or 16 in 3D space by our proposed method using (6) or by the de Casteljau algorithm. From the table we can see that the time costs for both methods are approximately proportional to the numbers of points. The time costs for computing the coefficient matrices of linear differential systems are also given in the table. As the time costs for computing the matrices are very small, they can even be ignored when a large number of points are evaluated by the dynamic approach.

TABLE 2
Time costs for generating Bézier curves of degree 3, 5, 8, or 16 (seconds).

Method	Points	Degree = 3	Degree = 5	Degree = 8	Degree = 16
de Casteljau	1×10^6	0.328	0.406	0.578	1.406
Dynamic	1×10^6	0.078	0.109	0.250	0.796
de Casteljau	2×10^6	0.656	0.813	1.188	2.812
Dynamic	2×10^6	0.125	0.219	0.453	1.562
coef-matrix		1.090E-6	2.190E-6	5.620E-6	3.021E-5

Example 2. Similar to Bézier curves, free-form curves defined by trigonometric or hyperbolic functions can be described by control points together with normalized B-basis [31, 33] or cyclic basis [30]. We show here the dynamic generation of Bézier-like curves defined by trigonometric functions. Other control point based curves can be generated in the same way. As discussed in [31], for all fixed values of the shape parameter $\beta \in (0, \pi)$ the unique nonnegative normalized B-basis of order n (degree $2n$) of the vector space

$$\mathbb{T}_{2n}^{0,\beta} = \text{span}\{1, \sin u, \cos u, \dots, \sin(nu), \cos(nu) : u \in [0, \beta]\}$$

is $\{T_{i,2n}^{\beta}(u) = t_{i,2n}^{\beta} \sin^i\left(\frac{u}{2}\right) \sin^{2n-i}(\frac{\beta-u}{2}), u \in [0, \beta]\}_{i=0}^{2n}$, where $\{t_{i,2n}^{\beta}\}_{i=0}^{2n}$ denote the nonnegative normalizing coefficients

$$t_{i,2n}^{\beta} = t_{2n-i,2n}^{\beta} = \frac{1}{\sin^{2n}\left(\frac{\beta}{2}\right)} \sum_{r=0}^{\lfloor \frac{i}{2} \rfloor} \binom{n}{i-r} \binom{i-r}{r} \left(2\cos\left(\frac{\beta}{2}\right)\right)^{i-2r}, i = 0, 1, \dots, n.$$

For notational simplicity, in the following we use t_i to represent the normalizing coefficients when they are used to construct a linear differential system.

Suppose that a Bézier-like curve is defined by $X(u) = \sum_{i=0}^{2n} X_i T_{i,2n}^{\beta}(u)$, $u \in [0, \beta]$, where X_i, $i = 0, 1, \dots, 2n$, are the given or the lifted control points in $\mathbb{R}^{2n+1}$. Let $M_X = (X_0, X_1, \dots, X_{2n})$ and $\Phi(u) = (T_{0,2n}^{\beta}(u), T_{1,2n}^{\beta}(u), \dots, T_{2n,2n}^{\beta}(u))^T$. The curve can then be reformulated as $X(u) = M_X \Phi(u)$. Based on [31], the derivative of the basis vector is $\frac{d\Phi(u)}{du} = C_n^{\beta} \Phi(u)$, where

$$C_n^{\beta} = \begin{pmatrix} -\frac{n}{\tan\frac{\beta}{2}} & -\frac{t_0}{t_1}\frac{2n}{2\sin\frac{\beta}{2}} & 0 & \cdots & 0 & 0 \\ \frac{t_1}{t_0}\frac{1}{2\sin\frac{\beta}{2}} & -\frac{n-1}{\tan\frac{\beta}{2}} & -\frac{t_1}{t_2}\frac{2n-1}{2\sin\frac{\beta}{2}} & \cdots & 0 & 0 \\ 0 & \frac{t_2}{t_1}\frac{2}{2\sin\frac{\beta}{2}} & -\frac{n-2}{\tan\frac{\beta}{2}} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{n-1}{\tan\frac{\beta}{2}} & -\frac{t_{2n-1}}{t_{2n}}\frac{1}{2\sin\frac{\beta}{2}} \\ 0 & 0 & 0 & \cdots & \frac{t_{2n}}{t_{2n-1}}\frac{2n}{2\sin\frac{\beta}{2}} & \frac{n}{\tan\frac{\beta}{2}} \end{pmatrix}.$$

Thus, the linear differential system that represents the curve $X(u)$ is

$$\text{(10)} \qquad \begin{cases} X'(u) = AX(u), & u \in [0, \beta], \\ X(0) = X_0, \end{cases}$$

where $A = M_X C_n^{\beta} M_X^{-1}$.

From the construction of the linear differential system (10) we can see that if the values of $\sin\frac{\beta}{2}$, $\cos\frac{\beta}{2}$, and $\tan\frac{\beta}{2}$ are known, all entries of the tridiagonal matrix C_n^{β}

and the coefficient matrix A will be computed by elementary arithmetic operations. In our experiments we compute the values of $\sin\frac{\beta}{2}$, $\cos\frac{\beta}{2}$, and $\tan\frac{\beta}{2}$ by loading the mathematical library functions, and it takes 6.250×10^{-6} or 3.140×10^{-5} seconds to compute the coefficient matrix for a differential system that represents a free-form curve of degree 8 or degree 16. When a differential system is established, we no longer need any computation of transcendental functions for generating a trigonometric curve and the time complexity for solving a differential system is the same as that for generating a Bézier curve of the same degree. We note that any point on the curve $X(u)=\sum_{i=0}^{2n}X_iT_{i,2n}^{\beta}(u)$ can also be evaluated by a Horner algorithm which has a linear time complexity in general, but transcendental functions have to be evaluated to generate a point.

Example 3. In the third example we generate a planar curve defined by a given curvature radius function. Unlike curves defined by normalized B-bases, an intrinsically defined planar curve can have an unlimited parameter domain [35]. Suppose that $\rho(t)$ represents the curvature radius of a planar curve and θ is the angle between the tangent direction of the curve and the positive direction of x-axis. The Cartesian coordinates of the intrinsically defined curve are given by

$$\mathbf{r}(\theta)=\begin{pmatrix}x(\theta)\\ y(\theta)\end{pmatrix}=\begin{pmatrix}\int_0^\theta\rho(t)\cos t dt\\ \int_0^\theta\rho(t)\sin t dt\end{pmatrix}. \tag{11}$$

In particular, we choose $\rho(t)=at^3+bt^2+ct+d$. The curve is

$$\begin{aligned}\mathbf{r}(\theta)=&\begin{pmatrix}6a-c\\ -2b+d\end{pmatrix}+\begin{pmatrix}-2b+d\\ -6a+c\end{pmatrix}\sin\theta+\begin{pmatrix}-6a+c\\ 2b-d\end{pmatrix}\cos\theta\\ &+\begin{pmatrix}-6a+c\\ 2b\end{pmatrix}\theta\sin\theta+\begin{pmatrix}2b\\ 6a-c\end{pmatrix}\theta\cos\theta+\begin{pmatrix}b\\ 3a\end{pmatrix}\theta^2\sin\theta\\ &+\begin{pmatrix}3a\\ -b\end{pmatrix}\theta^2\cos\theta+\begin{pmatrix}a\\ 0\end{pmatrix}\theta^3\sin\theta+\begin{pmatrix}0\\ -a\end{pmatrix}\theta^3\cos\theta.\end{aligned}$$

Using the same technique as discussed in section 2.2, we lift the curve from $\mathbb{R}^2$ to $\mathbb{R}^9$. Denote the lifted curve as $X(\theta)$ with new coefficient vectors $X_i\in\mathbb{R}^9$, $i=0,1,\ldots,8$; we construct a matrix of order 9 as $M_X=(X_0,X_1,\ldots,X_8)$. Furthermore, a differential of the basis vector with respect to the parameter θ is

$$\frac{d}{d\theta}\begin{pmatrix}1\\ \sin\theta\\ \cos\theta\\ \theta\sin\theta\\ \theta\cos\theta\\ \theta^2\sin\theta\\ \theta^2\cos\theta\\ \theta^3\sin\theta\\ \theta^3\cos\theta\end{pmatrix}=\begin{pmatrix}0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0\\ 0&-1&0&0&0&0&0&0&0\\ 0&1&0&0&1&0&0&0&0\\ 0&0&1&-1&0&0&0&0&0\\ 0&0&0&2&0&0&1&0&0\\ 0&0&0&0&2&-1&0&0&0\\ 0&0&0&0&0&3&0&0&1\\ 0&0&0&0&0&0&3&-1&0\end{pmatrix}\begin{pmatrix}1\\ \sin\theta\\ \cos\theta\\ \theta\sin\theta\\ \theta\cos\theta\\ \theta^2\sin\theta\\ \theta^2\cos\theta\\ \theta^3\sin\theta\\ \theta^3\cos\theta\end{pmatrix}.$$

Denote the coefficient matrix of above equation as C_ρ and let $A=M_XC_\rho M_X^{-1}$. A linear differential system for representing the curve $X(\theta)$ is

$$\begin{cases}X'(\theta)&=&AX(\theta),\qquad\theta\in[0,+\infty),\\ X(0)&=&X_0+X_2.\end{cases} \tag{12}$$

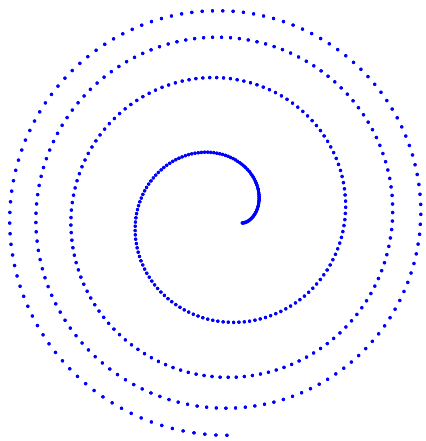


FIG. 1. *Dynamic generation of an intrinsically defined planar curve with* $\rho(t) = 0.001t^3 - 0.06t^2 + 1.5t + 0.4$*,* $t \in [0, 8\pi]$.

We choose $a = 0.001$, $b = -0.06$, $c = 1.5$, and $d = 0.4$ for defining the integral curve and generate a curve segment with $\theta \in [0, 8\pi]$. When the differential system (12) is established, we employ the Taylor method (with $s = 5$) to solve the differential system. Figure 1 illustrates a sequence of points computed by the proposed method with time step $\Delta\theta = \frac{8\pi}{500}$. To measure the accuracy of the proposed method, we compute the Euclidean distance between the last point generated by the proposed method and the point $\mathbf{r}(8\pi)$ which is computed by loading the mathematical library functions. The deviation distance for the last point is 1.766×10^{-7}, 5.552×10^{-9}, or 1.709×10^{-10} when the time step is chosen as $\Delta\theta = \frac{8\pi}{500}$, $\Delta\theta = \frac{8\pi}{1000}$, or $\Delta\theta = \frac{8\pi}{2000}$. Since the coefficient matrix of differential system (12) has the same order as that of differential system (9), which represents a Bézier curve of degree 8, the time costs for computing the coefficient matrix or generating the integral curve by solving the differential system are almost the same with that for generating a Bézier curve of degree 8. We note that for this example evaluating points on the integral curve by loading the mathematical library functions and using a lookup table takes about one-third of time costs needed by our proposed evaluation method. Nevertheless, the new approach benefits from using only arithmetic operations during the whole evaluation process. As no special library functions are needed for solving the linear differential system, our proposed method can be used to evaluate intrinsically defined curves on general computing machines.

Example 4. The fourth example is about generating a circle in 3D space. Suppose that $\mathbf{n}$, U, and V are unit vectors in $\mathbb{R}^3$ that are perpendicular to each other. A circle centered at $O = d_0\mathbf{n}$ with radius $r > 0$ is given by $X(t) = O + rU\cos t + rV\sin t$, $t \in [0, 2\pi]$. Since the space $\Omega = \text{span}\{1, \cos t, \sin t\}$ is closed under differentiation with respect to t, the differential system of which the circle is the solution curve is

$$\text{(13)} \qquad \begin{cases} X'(t) = AX(t), & t \in [0, 2\pi], \\ X(0) = d_0\mathbf{n} + rU. \end{cases}$$

In particular, the coefficient matrix is

$$A = (d_0\mathbf{n}, rU, rV) \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} (d_0\mathbf{n}, rU, rV)^{-1}$$

when $d_0 \neq 0$ or $A = VU^T - UV^T$ otherwise. It is easily verified that the coefficient matrix is skew-symmetric for both cases.

To guarantee that the evaluated points lie on the circle exactly, we employ the generalized implicit midpoint scheme discussed in section 3 to solve the differential system (13). To compute K points uniformly on the circle we choose $\Delta t = \frac{2\pi}{K}$ and $\frac{1}{2}h(\Delta t) = \tan(\frac{\Delta t}{2})$. The value of $\frac{1}{2}h(\Delta t)$ is further evaluated by the Taylor expansion of $\tan(\frac{\Delta t}{2})$ as $\frac{\Delta t}{2} + \frac{\Delta t^3}{24} + \frac{\Delta t^5}{240} + \frac{17\Delta t^7}{40320}$. Substituting this value of $\frac{1}{2}h(\Delta t)$ into (8), we generate all K sampled points on the circle from the given beginning point. Note that all these points are computed just by elementary arithmetic operations.

Concretely, we choose $d_0 = 0.5$ and $r = 1.0$ for defining a circle by (13) and the number of the evaluated points K varies between 100 and 1000. A set of points have then been generated by applying the generalized implicit midpoint scheme or the Taylor method to solve the differential system (13). For comparison, we also evaluate the same set of points from the circle by direct computation of the functions $\sin t$ and $\cos t$. Since Y_0 and Y_K are theoretically identical, we use the distance between Y_0 and Y_K as an accuracy measure of various evaluation methods. From the experiments we know that the evaluation accuracy by the generalized implicit midpoint scheme or by the direct computation of the transcendental functions does not depend much on the sampling step; the norms $\|Y_K - Y_0\|$ by these two methods are around 10^{-13} for various choices of K. Nevertheless, the generalized implicit mid-point scheme is much faster than direct evaluation of the transcendental functions. It takes 0.265 seconds to compute 10^7 points by the proposed method, while the direct approach needs 0.703 seconds. When we solve the differential system (13) by the Taylor method (with $s = 7$), the accuracy evidently depends on the time step. The absolute distance between Y_0 and Y_K is 1.557×10^{-7} when $K = 100$ and the distance reduces to 1.554×10^{-11} when $K = 1000$.

Example 5. The last example is about evaluating the Zhukovsky profile (also known as the airfoil profile), which is represented by a closed rational trigonometric curve [14]. Let $z = x + iy$ be a complex number. The Zhukovsky profile is generated by the conformal mapping $z + \frac{1}{z}$ of a circle that is centered at $(C_x, C_y)^T$ with radius r such that it passes through the point $(1, 0)^T$ and contains the point $(-1, 0)^T$. Suppose that

$$\begin{cases} a_1 = r^2 C_x, \\ b_1 = r^2 C_y, \\ c_1 = r(3C_x^2 + C_y^2 + r^2 + 1), \\ d_1 = 2rC_xC_y, \\ e_1 = C_x(C_x^2 + C_y^2 + 2r^2 + 1), \end{cases} \quad \begin{cases} a_2 = -r^2 C_y, \\ b_2 = r^2 C_x, \\ c_2 = 2rC_xC_y, \\ d_2 = r(C_x^2 + 3C_y^2 + r^2 - 1), \\ e_2 = C_y(C_x^2 + C_y^2 + 2r^2 - 1), \end{cases}$$

and

$$\begin{cases} c_3 = 2rC_x, \\ d_3 = 2rC_y, \\ e_3 = C_x^2 + C_y^2 + r^2. \end{cases}$$

The Cartesian coordinates of the Zhukovsky profile are computed by the rational functions

$$\begin{cases} x(t) = \frac{x_1(t)}{x_3(t)}, \\ y(t) = \frac{x_2(t)}{x_3(t)}, \end{cases} \tag{14}$$

where

$$\begin{cases} x_1(t) = a_1 \cos 2t + b_1 \sin 2t + c_1 \cos t + d_1 \sin t + e_1, \\ x_2(t) = a_2 \cos 2t + b_2 \sin 2t + c_2 \cos t + d_2 \sin t + e_2, \qquad t \in [0, 2\pi]. \\ x_3(t) = c_3 \cos t + d_3 \sin t + e_3, \end{cases}$$

Since the space $\Omega = \text{span}\{1, \cos t, \sin t, \cos 2t, \sin 2t\}$ is closed under differentiation with respect to t, the homogeneous coordinates $x_1(t), x_2(t), x_3(t)$ of the curve form the solutions of a differential system. By lifting the homogeneous coordinates of the curve to $\mathbb{R}^5$, the curve can be represented by

$$X(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 & d_1 & e_1 \\ a_2 & b_2 & c_2 & d_2 & e_2 \\ 0 & 0 & c_3 & d_3 & e_3 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \cos 2t \\ \sin 2t \\ \cos t \\ \sin t \\ 1 \end{pmatrix}.$$

Denote the coefficient matrix of the above equation as M_X and let

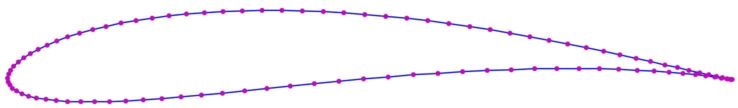
$$A = M_X \begin{pmatrix} 0 & -2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} M_X^{-1}.$$

The linear differential system that represents the lifted curve $X(t)$ is

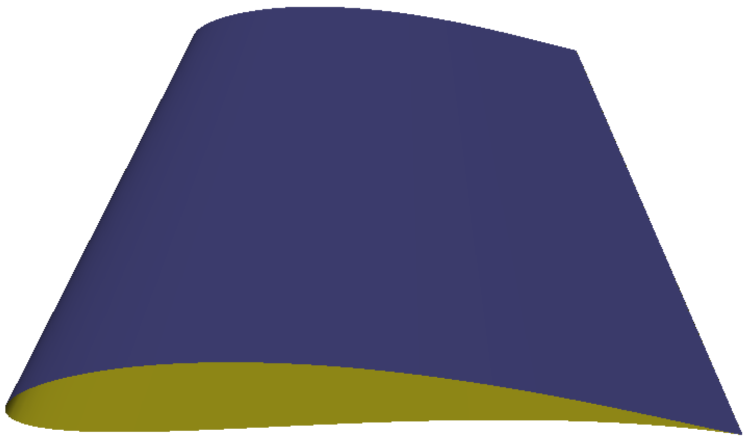
$$\text{(15)} \qquad \begin{cases} X'(t) = AX(t), \qquad t \in [0, 2\pi], \\ X(0) = \begin{pmatrix} a_1 + c_1 + e_1 \\ a_2 + c_2 + e_2 \\ c_3 + e_3 \\ 1 \\ 0 \end{pmatrix}. \end{cases}$$

As in [14] we choose $r = 1.1$, $C_y = 0.1$, and $C_x = 1.0 - \sqrt{1.1^2 - 0.1^2}$. The differential system (15) is solved numerically using the Taylor method. By choosing the coordinates x_1, x_2, and x_3 and applying (14) we generate the Zhukovsky profile. As points $(x(0), y(0))^T$ and $(x(2\pi), y(2\pi))^T$ are theoretically identical, we measure the accuracy of the numerical results using the distance between the starting point and the last point when a closed curve is generated. The evaluation accuracy using the Taylor algorithm ($s = 6$) is 1.564×10^{-3}, 1.581×10^{-9}, or 3.164×10^{-13} when the time step is chosen as $\Delta t = \frac{2\pi}{10}$, $\Delta t = \frac{2\pi}{100}$, or $\Delta t = \frac{2\pi}{1000}$. Figure 2(a) illustrates the plotted curve with 100 evaluated points and Figure 2(b) shows a surface that is generated by a series of isoparametric curves.

5. Conclusions. This paper has shown that free-form curves with properly defined basis functions are the solution curves of linear differential systems. Points and derivatives of the free-form curves can then be evaluated by employing typical numerical methods to solve the differential systems. Not only can the numerical algorithms achieve high accuracy or even exact results for point and derivative evaluation, but the new method also benefits from computational efficiency and uses only simple arithmetic operations for generating free-form curves and surfaces that are defined by algebraic as well as transcendental functions.



(a)



(b)

FIG. 2. *Dynamic generation of airfoil profile and airfoil surface:* (a) *the profile together with the evaluated points; and* (b) *the airfoil surface generated from a series of isoparametric curves.*

Acknowledgment. We owe thanks to referees for their invaluable comments and suggestions which helped to improve the presentation of the paper greatly.

REFERENCES

[1] P. E. Bézier, *Numerical Control: Mathematics and Applications*, Wiley, New York, 1972.

[2] M. Brilleaud and M. Mazure, *Mixed hyperbolic/trigonometric spaces for design*, Comput. Math. Appl., 64 (2012), pp. 2459–2477.

[3] Q. Chen and G. Wang, *A class of Bézier-like curves*, Comput. Aided Geom. Design, 20 (2003), pp. 29–39.

[4] J. Delgado and J. M. Peña, *A corner cutting algorithm for evaluating rational Bézier surfaces and the optimal stability of the basis*, SIAM J. Sci. Comput., 29 (2007), pp. 1668–1682.

[5] G. Farin, *NURBS: From Projective Geometry to Practical Use*, 2nd ed., A. K. Peters, Natick, MA, 1999.

[6] G. Farin, *Curves and Surfaces for CAGD: A Practical Guide*, 5th ed., Morgan Kaufmann, Burlington, MA, 2001.

[7] A. R. Forrest, *Interactive interpolation and approximation by Bézier polynomials*, Computer J., 15 (1972), pp. 71–79.

[8] S. Y. Gatilov, *Vectorizing NURBS surface evaluation with basis functions in power basis*, Comput. Aided Design, 73 (2016), pp. 26–35.

[9] Q. J. Ge, L. Srinivasan, and J. Rastegar, *Low-harmonic rational Bézier curves for trajectory generation of high-speed machinery*, Comput. Aided Geom. Design, 14 (1997), pp. 251–271.

[10] R. Goldman, *The fractal nature of Bézier curves*, in Proceedings Geometric Modeling and Processing (GMP 2004), Theory and Applications, Beijing, China, 2004, pp. 3–11.

[11] R. Goldman, *An Integrated Introduction to Computer Graphics and Geometric Modeling*, CRC Press, Boca Raton, FL, 2009.

[12] D. Gonsor and M. Neamtu, *Non-polynomial polar forms*, in Curves and Surfaces II, P. J. Laurent, A. LéMehauté, and L. L. Schumaker, eds., A. K. Peters, Wellesley, MA, 1994, pp. 193–200.

[13] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*, 2nd ed., Springer, New York, 2005.
[14] I. Juhász and Á. Róth, *Closed rational trigonometric curves and surfaces*, J. Comput. App. Math., 234 (2010), pp. 2390–2404.
[15] I. Koren and O. Zinaty, *Evaluating elementary functions in a numerical coprocessor based on rational approximations*, in Proc. IEEE Trans. Computers, 39 (1990), pp. 1030–1037.
[16] Y. Li, *On the recurrence relations for B-splines defined by certain L-splines*, J. Approx. Theory, 43 (1985), pp. 359–369.
[17] Y. Li and G. Wang, *Two kinds of B-basis of the algebraic hyperbolic space*, J. Zhejiang Univ. Sci. A, 6 (2005), pp. 750–759.
[18] S. Lien, M. Shantz, and V. R. Pratt, *Adaptive forward differencing for rendering curves and surfaces*, in Proceedings of SIGGRAPH, 1987, pp. 111–118.
[19] Y. Lü, G. Wang, and X. Yang, *Uniform hyperbolic polynomial B-spline curves*, Comput. Aided Geom. Design, 19 (2002), pp. 379–393.
[20] W. L. Luken and F. Cheng, *Comparison of surface and derivative evaluation methods for the rendering of NURB surfaces*, ACM Trans. Graph., 15 (1996), pp. 153–178.
[21] E. Mainar and J. M. Peña, *Quadratic-cycloidal curves*, Adv. Comput. Math., 20 (2004), pp. 161–175.
[22] E. Mainar, J. M. Peña, and J. Sánchez-Reyes, *Shape preserving alternatives to the rational Bézier model*, Comput. Aided Geom. Design, 18 (2001), pp. 37–60.
[23] C. Moler and C. V. Loan, *Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later*, SIAM Rev., 45 (2003), pp. 3–49.
[24] G. Morin and R. Goldman, *A subdivision scheme for Poisson curves and surfaces*, Comput. Aided Geom. Design, 17 (2000), pp. 813–833.
[25] R. Nave, *Implementation of transcendental functions on a numerics processor*, Microprocessing Microprogramming, 11 (1983), pp. 221–225.
[26] J. M. Peña, *Shape preserving representations for trigonometric polynomial curves*, Comput. Aided Geom. Design, 14 (1997), pp. 5–11.
[27] L. Piegl and W. Tiller, *The NURBS Book*, 2nd ed., Springer, New York, 1997.
[28] Á. Róth, *Control point based exact description of curves and surfaces, in extended Chebyshev spaces*, Comput. Aided Geom. Design, 40 (2015), pp. 40–58.
[29] Á. Róth and I. Juhász, *Control point based exact description of a class of closed curves and surfaces*, Comput. Aided Geom. Design, 27 (2010), pp. 179–201.
[30] Á. Róth, I. Juhász, J. Schicho, and M. Hoffmann, *A cyclic basis for closed curve and surface modeling*, Comput. Aided Geom. Design, 26 (2009), pp. 528–546.
[31] J. Sánchez-Reyes, *Harmonic rational Bézier curves, p-Bézier curves and trigonometric polynomials*, Comput. Aided Geom. Design, 15 (1998), pp. 909–923.
[32] J. Sánchez-Reyes, *Periodic Bézier curves*, Comput. Aided Geom. Design, 26 (2009), pp. 989–1005.
[33] W. Shen and G. Wang, *A class of quasi Bézier curves based on hyperbolic polynomials*, J. Zhejiang Univ. Sci. A, 6 (2005), pp. 116–123.
[34] W. Shen and G. Wang, *Geometric shapes of C-Bézier curves*, Comput. Aided Design, 58 (2015), pp. 242–247.
[35] W. Wu and X. Yang, *Geometric Hermite interpolation by a family of intrinsically defined planar curves*, Comput. Aided Design, 77 (2016), pp. 86–97.
[36] G. Xu and G. Wang, *AHT Bézier curves and NUAH B-spline curves*, J. Comput. Sci. Technol., 22 (2007), pp. 597–607.
[37] J. Zhang, *C-curves: An extension of cubic curves*, Comput. Aided Geom. Design, 13 (1996), pp. 199–217.