VANDERMONDE FACTORIZATIONS OF A REGULAR HANKEL MATRIX AND THEIR APPLICATION TO THE COMPUTATION OF BÉZIER CURVES

LICIO H. BEZERRA*

Abstract. In this paper each coordinate of a Bézier curve $B(s)$ of degree $(n-1)$, $n = 2m-1$, is expressed as a Hankel form applied to the vector $(B_m^e(s))^T e_m$, where $B_m^e(s)$ is the $m \times m$ Bernstein matrix and e_m is the mth canonical vector of $\mathbb{C}^m$. These expressions can be easily calculated once we have a Vandermonde factorization of the Hankel matrices associated to the forms. To that end, we begin presenting another proof of the existence of a Vandermonde factorization of a regular Hankel matrix by using Pascal matrices techniques. In the cases where one or both Hankel matrices associated to the forms are ill-conditioned with respect to inversion, we propose shifting their skew-diagonals and counteracting them after, which is done practically without computational costs. By comparing this way of computing a Bézier curve with other current methods, we see that the results suggest that this approach is very promising with regard to accuracy and time of computation, even for large values of n. Examples of the behavior of this kind of method under degree elevation and degree reduction are also presented here.

Key words. Pascal matrix, Bernstein matrix, Bézier curve, Hankel form, Vandermonde factorization

AMS subject classifications. 12E10, 15A23, 15B05, 65D17

1. Introduction. Suppose we have a Bézier curve B of degree $n-1$ defined by n given points $Q_0 = (x_0, y_0)$, $Q_1 = (x_1, y_1)$, ..., $Q_{n-1} = (x_{n-1}, y_{n-1})$ in $\mathbb{R}^2$. That is,

$$B(s) = (x(s), y(s)) = \sum_{i=0}^{n-1} Q_i b_{i,n-1}(s), \qquad s \in [0,1],$$

where $b_{i,n-1}(s) = \binom{n-1}{i} s^i (1-s)^{n-1-i}$ for each $i \in \{0, ..., n-1\}$. Let $B_n^e(s)$ be the $n \times n$ lower triangular matrix such that $(B_n^e)_{ij}(s) = b_{j-1,i-1}(s)$ for each $n \geq i \geq j \geq 1$. $B_n^e(s)$ is called a Bernstein matrix [1]. We note that the Bernstein polynomials $b_{0,n-1}(s), ..., b_{n-1,n-1}(s)$ form a basis of $\mathbb{P}_{n-1}(\mathbb{R})$, the vector space of the real polynomial functions of degree less than or equal to $(n-1)$.

Paul de Casteljau developed a very stable algorithm to evaluate Bézier curves [7]. Since then, several algorithms for polynomial evaluation have been proposed, which can also be used to evaluate Bézier curves. A polynomial curve C of degree $n-1$ defined by n control points $Z_0, ..., Z_{n-1}$ is described as follows:

$$C(s) = \sum_{i=0}^{n-1} Z_i w_{i,n-1}(s),$$

where $\{w_{0,n-1}(s), ..., w_{n-1,n-1}(s)\}$ is some basis of $\mathbb{P}_{n-1}(\mathbb{R})$. For instance, the VS algorithm ([22]) uses the basis $\{v_{0,n-1}(s), ..., v_{n-1,n-1}(s)\}$, where $v_{i,n-1}(s) = s^i(1-s)^{n-1-i}$ for each $i \in \{0, ..., n-1\}$. Other examples are the Wang-Ball algorithm ([24]), the Said-Ball algorithm ([24]), and the DP-Ball algorithm ([12]). All of these use higher degree generalizations of the Ball basis – the set $\{(1-s)^2, 2s(1-s)^2, 2s^2(1-s), s^2\}$, which was introduced in the seventies for geometric modeling ([2]). In order

*Departamento de Matemática, Universidade Federal de Santa Catarina, Florianópolis, SC, Brazil 88040-900 (licio@mtm.ufsc.br).

to evaluate a Bézier curve we first make a change of coordinates from the Bernstein system to one of these systems. Then we use the respective evaluation algorithm to compute points on the curve (see [21]). Lately, Bézier curves have also been computed from matrix theory. For instance, in [4] Bézier curves of degree $n-1$ were computed in $\mathcal{O}(n \log n)$ operations from the expression of an $n \times n$ Bernstein matrix in terms of the lower triangular Pascal matrix P_n (see [1]). However, the performance of these so-called Pascal matrix methods becomes very unstable when n attains some level, which depends on the computer architecture. In order to overcome the limitation on the degree of the curve for Pascal matrix methods we created a new algorithm. This algorithm is based on a different way of formulating Bézier curves, which involves Hankel matrices.

Let H be a Hankel matrix of order n, i.e., $(\forall i, j \in \{1, ..., n\})\ H_{ij} = h_{i+j-1}$. A very known theorem states that, if H is nonsingular, then a Vandermonde matrix V and a diagonal matrix D exist so that $H = VDV^T$. There is a proof of this fact in [19], which utilizes a class of matrices arisen in the theory of root separation of algebraic polynomials, namely the class of Bezoutians. A different proof of this theorem appears in [16] by using classical results of matrix theory. Here we present another proof of this theorem in §2. To that end, we first obtain some preliminary results by utilizing Pascal matrices techniques. Then, if x_γ is the solution of $Hx = (h_{n+1} \dots h_{2n-1}\, \gamma)^T$, $\gamma \in \mathbb{C}$, we will see that the existence of a Vandermonde factorization of a regular Hankel matrix H depends on the non-existence of a multiple eigenvalue of the companion matrix $C_\gamma = compan(x_\gamma)$ for some complex number γ. Here, $compan(x)$ denotes the matrix $[e_2^T; e_3^T; ...; e_n^T; x^T]$, where $\{e_1, ..., e_n\}$ is the canonical basis of $\mathbb{C}^n$ and $x \in \mathbb{C}^n$. An interesting result is that C_γ has no multiple eigenvalue for all but a finite set of complex numbers γ. Thus, it is possible to create a general algorithm that computes a Vandermonde factorization of a nonsingular Hankel matrix. Note that even when γ belongs to that finite set the following factorization of H from C_γ is still possible: $H = V_c D V_c^T$, where V_c is a confluent Vandermonde matrix and D, a block diagonal matrix [8]. From now on we will use the MATLAB notation $vander([\alpha_1, ..., \alpha_n])$ to denote the following Vandermonde matrix:

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \cdots & \vdots \\ \alpha_1^{n-1} & \alpha_2^{n-1} & \cdots & \alpha_n^{n-1} \end{pmatrix},$$

which is different from the matrix generated by MATLAB with that command.

In §3, we see that we can express each coordinate of a Bézier curve of degree $n-1$, $n = 2m-1$, as a Hankel form on $\mathbb{C}^m$ applied to the vector $(B_m^e(s))^T e_m$, where $B_m^e(s)$ is the $m \times m$ Bernstein matrix and e_m is the mth canonical vector of $\mathbb{C}^m$. Still in this section we propose two algorithms to compute Bézier curves from Vandermonde factorizations of the associated Hankel matrices. In §4, some results of numerical experiments are presented. We see that we can compute Bézier curves by our algorithms in a very fast and precise way, which is corroborated from the comparisons done with other methods. Algorithm 1 computes the eigenvalues of $H^{-1}H_1$, where H_1 is the Hankel matrix whose first column is $(\, h_2 \dots h_{n+1})^T$ and whose last row is $(\, h_{n+1} \dots \gamma\,)$; Algorithm 2 computes them from the equivalent generalized eigenvalue problem $H_1 x = \lambda H x$. On one hand the computation of these eigenvalues certainly depends on a good value for γ. On the other hand, several experiments have indicated that the computation of these eigenvalues is very sensitive to its condition

with respect to inversion, and this is corroborated by some theoretical results (e.g, see Corollary 3.1 of [3]). We have modified our methods to handle an ill-conditioned Hankel matrix by shifting its skew-diagonal entries toward skew-diagonal dominance. The experiments with this procedure applied to Algorithm 1 have shown that the precision of all the computation improves. Furthermore, we have seen that it is not expensive with respect to time. We finish the section introducing a subdivision formula in terms of some Hankel matrices, presenting some examples to illustrate the behavior of Algorithm 1 under degree elevation and degree reduction, and introducing a new corner cutting system on $[0,1]$ for Bézier curves of degree 3. This system is made up of functions of the type our method uses in order to compute such curves.

2. Vandermonde factorizations of a nonsingular Hankel matrix.

In this section we present a proof of the existence of a Vandermonde factorization of a nonsingular Hankel matrix H, $H = hankel([h_1, ..., h_m], [h_m, ..., h_{2m-1}])$. That is,

$$H = \begin{pmatrix} h_1 & h_2 & ... & h_m \\ \vdots & \vdots & \vdots & \vdots \\ h_{m-1} & h_m & ... & h_{2m-2} \\ h_m & h_{m+1} & ... & h_{2m-1} \end{pmatrix}.$$

In order to obtain a Vandermonde matrix as a factor of H, we first compute x_γ, which is the solution of $Hx = y_\gamma$, where $y_\gamma = (h_{m+1} \, ... \, h_{2m-1} \, \gamma)^T$, and then we compute the spectrum of the companion matrix $C_\gamma = [e_2^T; ...; e_m^T; x_\gamma^T]$. Note that such a matrix is a non-derogatory matrix, that is, a matrix whose characteristic polynomial and minimum polynomial are identical. This means that, if a non-derogatory matrix has an eigenvalue whose algebraic multiplicity is greater that 1, then the matrix is not diagonalizable. The goal of this section is to show that C_γ is diagonalizable for all but a finite set of complex numbers γ. This fact allows a computer to generate γ by itself.

We begin presenting some algebraic properties of Hankel matrices. To this end, let $a = (a_0 \, ... \, a_{m-1})^T$ be the solution of $Hx = e_m$, and let $b = (b_0 \, ... \, b_{m-1})^T$ be the solution of $Hx = (h_{m+1} \, ... \, h_{2m-1} \, 0)^T$. Let $p_\gamma(x)$ be the characteristic polynomial of C_γ. Then $p_\gamma(x) = r(x) - \gamma s(x)$, where $r(x) = x^m - b_{m-1}x^{m-1} - ... - b_1 x - b_0$ and $s(x) = a_{m-1}x^{m-1} + ... + a_1 x + a_0$.

LEMMA 2.1. *Let H be an $m \times m$ nonsingular Hankel matrix. Let $a = (a_0 ... a_{m-1})^T$ and $b = (b_0 ... b_{m-1})^T$ be the solutions of $Hx = e_m$ and $Hx = (h_{m+1} \, ... \, h_{2m-1} \, 0)^T$, respectively. Then $a_0 \neq 0$ or $b_0 \neq 0$.*

Proof.

Suppose $|H(1 : m-1, 2 : m)| \neq 0$. Therefore, from Cramer's rule, $a_0 \neq 0$. Let $x_1, ..., x_{m-1}$ be the unique scalars such that

$$x_1 \begin{pmatrix} h_2 \\ \vdots \\ h_m \end{pmatrix} + ... + x_{m-1} \begin{pmatrix} h_m \\ \vdots \\ h_{2m-2} \end{pmatrix} = \begin{pmatrix} h_{m+1} \\ \vdots \\ h_{2m-1} \end{pmatrix}.$$

Hence, $x = (0 \, x_1 \, ... \, x_{m-1})^T = \gamma a + b$ is the solution of $Hx = (h_{m+1} \, ... \, h_{2m-1} \, \gamma)^T$ iff $\gamma = x_1 h_{m+1} + ... + x_{m-1} h_{2m-1}$. So, if γ is different from this value, the first coordinate x_0 of x is not zero. Since $x_0 = \gamma a_0 + b_0$, then we have $a_0 \neq 0$ or $b_0 \neq 0$. Observe that $a_0 \neq 0$ and $b_0 = 0$ iff $x_1 h_{m+1} + ... + x_{m-1} h_{2m-1} = 0$.

Now suppose $H(1:m-1,2:m) = H(2:m,1:m-1)$ is singular. Since H is nonsingular, the dimension of $span\{H(2:m,1),...,H(2:m,m-1),H(2:m,m)\}$ is $(m-1)$, as well as the dimension of $span\{H(1:m-1,1),...,H(1:m-1,m-1),H(1:m-1,m)\}$. Hence, $H(2:m,m) \notin span\{H(2:m,1),...,H(2:m,m-1)\}$, whose dimension is $m-2$. On the other side, $H(2:m,m) \in span\{H(1:m-1,1),...,H(1:m-1,m)\} = span\{H(1:m-1,1),H(2:m,1),...,H(2:m,m-1)\}$, and so, there is only one x_0, which is different from zero, such that

$$\begin{pmatrix} h_{m+1} \\ \vdots \\ h_{2m-1} \end{pmatrix} = x_0 \begin{pmatrix} h_1 \\ \vdots \\ h_{m-1} \end{pmatrix} + x_1 \begin{pmatrix} h_2 \\ \vdots \\ h_m \end{pmatrix} + ... + x_{m-1} \begin{pmatrix} h_m \\ \vdots \\ h_{2m-2} \end{pmatrix},$$

for some $x_1,...,x_{m-1}$. Observe in this case that $x_0 = b_0 \neq 0$ and $a_0 = 0$ for all $\gamma \in \mathbb{C}$. □

From the above proof, there can be at most one complex number γ such that $p_\gamma(0) = -b_0 - \gamma a_0 = 0$. We can also conclude from Lemma 2.1 that zero is not a common root of $r(x) = x^m - b_{m-1}x^{m-1} - ... - b_1x - b_0$ and $s(x) = a_{m-1}x^{m-1} + ... + a_1x + a_0$.

Now define $H_\gamma^\kappa = \begin{pmatrix} h_1 & ... & h_m & h_{m+1} \\ \vdots & \ddots & \vdots & \vdots \\ h_m & ... & h_{2m-1} & \gamma \\ h_{m+1} & ... & \gamma & \kappa \end{pmatrix}$. Since H is nonsingular, H_γ^κ is also nonsingular iff $\kappa \neq \kappa_0 = (h_{m+1}\ ...\ h_{2m-1}\ \gamma)\ H^{-1}(h_{m+1}\ ...\ h_{2m-1}\ \gamma)^T$, which is equal to $(h_{m+1}\ ...\ h_{2m-1}\ \gamma)(b_0 + \gamma a_0\ \ ...\ \ b_{m-2} + \gamma a_{m-2}\ \ b_{m-1} + \gamma a_{m-1})^T$.

Note that $H_\gamma^\kappa \begin{pmatrix} -b_0 - \gamma a_0 \\ \vdots \\ -b_{m-1} - \gamma a_{m-1} \\ 1 \end{pmatrix} = (\kappa - \kappa_0) \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$. So, Lemma 2.1 can be restated as the following lemma:

LEMMA 2.2. *Let* $H_\gamma^\kappa = \begin{pmatrix} h_1 & ... & h_m & h_{m+1} \\ \vdots & \ddots & \vdots & \vdots \\ h_m & ... & h_{2m-1} & \gamma \\ h_{m+1} & ... & \gamma & \kappa \end{pmatrix}$ *be a Hankel matrix, where* $H = H_\gamma^\kappa(1:m,1:m)$ *is nonsingular. Suppose that* H_γ^κ *is also nonsingular, that is,* $\kappa \neq (h_{m+1}\ ...\ h_{2m-1}\ \gamma)\ H^{-1}(h_{m+1}\ ...\ h_{2m-1}\ \gamma)^T$. *Let* p *be the solution of* $H_\gamma^\kappa x = e_{m+1}$. *Then* $p_0 \neq 0$ *except for one possible complex number* γ.

Let α be any complex number and let $q_\gamma(x) = p_\gamma(x+\alpha) = r(x+\alpha) - \gamma s(x+\alpha)$. It will be shown that $r(\alpha)$ and $s(\alpha)$ cannot be both null because there can be only one complex number γ such that $q_\gamma(0) = 0$. In order to prove this, some notations and definitions are introduced in the following.

DEFINITION 2.3. *Let* $\alpha \in \mathbb{C}$. *Let* $P_m[\alpha]$ *be the* $m \times m$ *lower triangular matrix*

defined for each $i, j \in \{1, 2, \ldots, m\}$ *by*

$$(P_m[\alpha])_{ij} = \begin{cases} \alpha^{i-j}\binom{i-1}{j-1} & , \quad \textit{for } i \geqslant j; \\ 0 & , \quad \textit{otherwise.} \end{cases}$$

$P_m[\alpha]$ *is said to be a generalized lower triangular Pascal matrix. If* $\alpha = 1$, $P_m[1] = P_m$ *is called the* $m \times m$ *lower triangular Pascal matrix.*

Some results about these matrices (see [1], [9]) are listed in the following lemma:

LEMMA 2.4. *Let* $P_m[\alpha]$ *be a generalized lower triangular Pascal matrix. Then*
(a) $P_m[0] = I_m$*;*
(b) $P_m[\alpha]P_m[\beta] = P_m[\alpha + \beta]$*;*
(c) $(P_m[\alpha])^{-1} = P_m[-\alpha]$*;*
(d) Suppose $\alpha \neq 0$ *and let* $G_m(\alpha)$ *be the* $m \times m$ *diagonal matrix such that* $(G_m(\alpha))_{kk} = \alpha^{k-1}$ *for all* $k \in \{1, ..., m\}$*. Then* $P_m[\alpha] = G_m(\alpha)P_mG_m(\alpha^{-1})$*. In particular,* $P_m^{-1} = G_m(-1)P_mG_m(-1)$*.*
(e) If $v = \left(1\, \alpha\, \alpha^2 \ldots \alpha^{m-1}\right)^T$*, then* $P_m^k v = \left(1\, (\alpha + k)\, (\alpha + k)^2 \ldots (\alpha + k)^{m-1}\right)^T$ *for all integer* k*.*

Recall the definition of the Bernstein matrix $B_m^e(s)$ for each scalar s:

$$[B_m^e(s)]_{ij} = \begin{cases} \binom{i-1}{j-1} s^{j-1}(1-s)^{i-j} & , \quad \text{for } m \geq i \geq j \geq 1; \\ 0 & , \quad \text{otherwise.} \end{cases}$$

A very important fact about Bernstein matrices that will be used here is the following proposition, whose proof can be found in [1]:

PROPOSITION 2.5. *Let* $s \in [0, 1]$ *and let* $B_m^e(s)$ *be an* $m \times m$ *Bernstein matrix Then,* $B_m^e(s) = P_mG_m(s)P_m^{-1}$*, where* P_m *is the* $m \times m$ *lower triangular Pascal matrix and* $G_m(s) = diag([1, s, ..., s^{m-1}])$*.*

In the following, we present some relations between Pascal and Hankel matrices.

LEMMA 2.6. *Let* H *be an* $m \times m$ *Hankel matrix and let* P_m *be the* $m \times m$ *lower triangular Pascal matrix. Then* $P_mHP_m^T$ *is still a Hankel matrix.*

Proof. The lemma obviously holds when $m = 1$. Suppose it holds for all Hankel matrices H of order $m \geq 1$. Now, let H be a $(m + 1) \times (m + 1)$ Hankel matrix and consider $P_{m+1}HP_{m+1}^T$. Since $P_{m+1}HP_{m+1}^T$ is symmetric and

$$P_{m+1}HP_{m+1}^T = \begin{pmatrix} P_mHP_m^T & v \\ v^T & \kappa \end{pmatrix}$$

for some $v \in \mathbb{C}^m$ and $\kappa \in \mathbb{C}$, it suffices to show that

$$(P_{m+1}HP_{m+1}^T)_{m+1,k} = (P_{m+1}HP_{m+1}^T)_{m,k+1}$$

for all $k \in \{1, ..., m - 1\}$. Observe that

$$(P_{m+1}HP_{m+1}^T)_{m+1,k} = e_{m+1}^T P_{m+1} \sum_{j=0}^{k-1} \binom{k-1}{j} He_{j+1} =$$

$$= \sum_{i=0}^{m}\sum_{j=0}^{k-1}\binom{m}{i}\binom{k-1}{j}e_{i+1}^T H e_{j+1} = \sum_{s=2}^{m+k-1} h_{s-1}\sum_{i=0}^{s}\binom{m}{i}\binom{k-1}{s-i}.$$

From Vandermonde convolution ([18]), this is equal to

$$\sum_{s=2}^{m+k-1} h_{s-1}\sum_{i=0}^{s}\binom{m-1}{i}\binom{k}{s-i} = \sum_{i=0}^{m-1}\sum_{j=0}^{k}\binom{m-1}{i}\binom{k}{j}e_{i+1}^T H e_{j+1} =$$

$$= e_m^T P_{m+1}\sum_{j=0}^{k}\binom{k}{j}He_{j+1} = (P_{m+1}HP_{m+1}^T)_{m,k+1}.$$

□

COROLLARY 2.7. *Let H be an $m\times m$ Hankel matrix. Then $P_m[\alpha]HP_m[\alpha]^T$ is still a Hankel matrix for any complex number α.*

Proof. For $\alpha = 0$ the result follows from Lemma 2.6. Let $\alpha \neq 0$. Frim Lemma 2.4, $P_m[\alpha] = G_m(\alpha)P_mG_m(\alpha^{-1})$, where $G_m(\alpha) = diag\,(1,\alpha,...,\alpha^{m-1})$. So, it suffices to show that $G_m(\alpha)HG_m(\alpha)$ is a Hankel matrix. But this is obviously true, for $(G_m(\alpha)HG_m(\alpha))_{ij} = h_{i+j-1}\alpha^{i+j-2}$. □

We see in the following proposition that $r(x)$ and $s(x)$ don't have any common root. The proof is done by using generalized Pascal matrix techniques.

PROPOSITION 2.8. *Let H be an $m\times m$ nonsingular Hankel matrix. Let $a = (a_0\, a_1\, ...\, a_{m-1})^T$ and $b = (b_0\, b_1\, ...\, b_{m-1})^T$ be the solutions of $Ha = e_m$ and $Hb = (h_{m+1}\, ...\, h_{2m-1}\, 0)^T$, respectively. Then $r(x) = x^m - b_{m-1}x^{m-1} - ... - b_1x - b_0$ and $s(x) = a_{m-1}x^{m-1} + ... + a_1x + a_0$ don't have any common root.*

Proof. Let $\gamma\in\mathbb{C}$ and let $p_\gamma = (-b_0 - \gamma a_0\, ...\, - b_{m-1} - \gamma a_{m-1}\, 1)^T$. Let $\alpha\in\mathbb{C}$ and let q_γ be the vector of coefficients of the polynomial $r(x+\alpha) - \gamma s(x+\alpha)$. We note that $q_\gamma = P_{m+1}[\alpha]^T p_\gamma = P_{m+1}[\alpha]^T(H_\gamma^\kappa)^{-1}e_{m+1}$ for $\kappa = 1+\kappa_0$. Thus, $H_\gamma^\kappa P_{m+1}[\alpha]^{-T}q_\gamma = e_{m+1}$, and so,

$$\widehat{H}_\gamma^\kappa q_\gamma = P_{m+1}[\alpha]^{-1}H_\gamma^\kappa P_{m+1}[\alpha]^{-T}q_\gamma = P_{m+1}[-\alpha]H_\gamma^\kappa P_{m+1}[-\alpha]^T q_\gamma = e_{m+1}.$$

Observe that $\widehat{H}_\gamma^\kappa$ is also nonsingular and, from corollary 2.7, is a Hankel matrix. Since $H_\gamma^\kappa = H_0^0 + \gamma\,(e_{m+1}e_m^T + e_me_{m+1}^T) + \kappa\, e_{m+1}e_{m+1}^T$, we see that $\widehat{H}_\gamma^\kappa = \widehat{H}_0^0 + \gamma\,(e_{m+1}e_m^T + e_me_{m+1}^T) + (\kappa - 2\,m\,\alpha)\,e_{m+1}e_{m+1}^T$. That is,

$$\widehat{H}_\gamma^\kappa = \begin{pmatrix} \hat{h}_1 & ... & \hat{h}_m & \hat{h}_{m+1} \\ \vdots & \ddots & \vdots & \vdots \\ \hat{h}_m & ... & \hat{h}_{2m-1} & \hat{\gamma} \\ \hat{h}_{m+1} & ... & \hat{\gamma} & \hat{\kappa} \end{pmatrix},$$

where $\widehat{H}_\gamma^\kappa(1:m,1:m) = \widehat{H} = P_m[-\alpha]HP_m[-\alpha]^T$ is nonsingular, and $\hat{\gamma} = \gamma + (\widehat{H}_0^0)_{m+1,m}$. Thus, from Lemma 2.2, $(q_\gamma)_0 \neq 0$ for some complex number γ. Since $q_\gamma)_0 = r(\alpha) - \gamma s(\alpha)$, we conclude that $r(\alpha)\neq 0$ or $s(\alpha)\neq 0$.

□

Note that $(q_\gamma)_0 = 0$ only if $s(\alpha) \neq 0$, which means that $\left|\widehat{H}^\kappa_\gamma(1:m-1, 2:m)\right| = \left|\widehat{H}(1:m-1, 2:m)\right| \neq 0$. In this case, $\gamma = r(\alpha)/s(\alpha)$.

PROPOSITION 2.9. *Let $\gamma \in \mathbb{C}$. Let $p_\gamma(x) = r(x) - \gamma s(x)$ be the characteristic polynomial of $C_\gamma = H_1(\gamma)H^{-1}$, where $H_1(\gamma)$ is the Hankel matrix defined by $H_1(\gamma)e_k = He_{k+1}$ for $k = 1, ..., m-1$, and $H_1(\gamma)e_m = (h_{m+1} \dots h_{2m-1}\, \gamma)^T$. Then the set of scalars γ such that C_γ is not diagonalizable is finite.*

Proof. $C_\gamma = [e_2^T; ...; e_m^T; (h_{m+1} \dots h_{2m-1}\, \gamma)H^{-1}]$ is a companion matrix. Hence, it is a non-derogatory matrix. That is, if C_γ has multiple eigenvalues, then C_γ is not diagonalizable. Thus, it suffices to show that the set of scalars γ for which C_γ has multiple eigenvalues is finite.

Let $\alpha \in \mathbb{C}$ be an eigenvalue of C_γ, that is, a root of $p_\gamma(x)$. Therefore, $r(\alpha) = \gamma s(\alpha)$. From Proposition 2.8 $s(\alpha) \neq 0$, and so there are two cases:

(i) $r(\alpha) = 0$ (this occurs iff $\gamma = 0$), and in this case C_0 is not diagonalizable iff $r'(\alpha) = 0$;

(ii) $r(\alpha) \neq 0$, which means that $\gamma = r(\alpha)/s(\alpha) \neq 0$. In this case, $p'_\gamma(\alpha) = 0$ iff either $r'(\alpha) = s'(\alpha) = 0$ or $r'(\alpha) = \gamma s'(\alpha) \neq 0$.

Therefore, since $s \neq 0$ and r/s is not a constant, α is contained in the set of the roots of $r's - rs'$, which has at most $2(m-1)$ elements. Hence, we conclude that $\{\gamma \in \mathbb{C} \,|\, C_\gamma \text{ is not diagonalizable}\}$ is finite and has at most $2(m-1)$ elements. ∎

We can now state the following theorem:

THEOREM 2.10. *Let H be an $m \times m$ nonsingular Hankel matrix. Let $r(x) = x^m - b_{m-1}x^{m-1} - ... - b_0$ and $s(x) = a_{m-1}x^{m-1} + ... + a_0$, where $a = (a_0\, a_1 \dots a_{m-1})^T$ and $b = (b_0\, b_1 \dots b_{m-1})^T$ satisfy $Ha = e_m$ and $Hb = (h_{m+1} \dots h_{2m-1}\, 0)^T$, respectively. Let $S = \{\, \alpha \in \mathbb{C} \,|\, (rs' - r's)(\alpha) = 0 \text{ and } s(\alpha) \neq 0 \,\}$ and $T = \{r(\alpha)/s(\alpha) \,|\, \alpha \in S\}$. Then, for all $\gamma \in \mathbb{C} - T$, $H = V_\gamma D_\gamma V_\gamma^T$, where $V_\gamma = vander(\alpha_1, ..., \alpha_m)$, $D_\gamma = diag\,(V_\gamma^{-1}He_1)$, $\{\, \alpha_1, ..., \alpha_m \,\} = \lambda(C_\gamma)$, and C_γ is the companion matrix whose last row is $(b_0 + \gamma a_0 \dots b_{m-1} + \gamma a_{m-1})$.*

Proof. From Proposition 2.9, $\lambda(C_\gamma)$ is simple for all $\gamma \in \mathbb{C} - T$. Suppose $\{\, \alpha_1, ..., \alpha_m \,\} = \lambda(C_\gamma)$. Let $v = (h_{m+1} \dots h_{2m-1}\, \gamma)^T$ and $H_1 = [H(2:m, :); v^T]$. Since H_1 is a symmetric matrix, $H_1 = [H(:, 2:m), v]$. Thus,

$$C_\gamma = H_1H^{-1} = V_\gamma\, diag([\alpha_1, ..., \alpha_m])\, V_\gamma^{-1},$$

where $V_\gamma e_i = \left(1\, \alpha_i \dots \alpha_i^{m-1}\right)^T$ for all $i \in \{1, ..., m\}$. So,

$$V_\gamma^{-1}H_1 = diag([\alpha_1, ..., \alpha_m])\, V_\gamma^{-1}H.$$

Let $d = (d_1 \dots d_m)^T = V_\gamma^{-1}He_1$. Let $D_\gamma = diag(d)$. Hence, for all $i \in \{1, ..., m-1\}$,

$$V_\gamma^{-1}He_{i+1} = V_\gamma^{-1}H_1e_i = diag([\alpha_1, ..., \alpha_m])\, V_\gamma^{-1}He_i =$$

$$= diag([\alpha_1, ..., \alpha_m])^i d = \left(d_1\alpha_1^i \dots d_m\alpha_m^i\right)^T = D_\gamma V_\gamma^T e_{i+1}.$$

Therefore, $H = V_\gamma D_\gamma V_\gamma^T$.

∎

Theoretically speaking, this theorem allows us to compute a Vandermonde factorization of a regular Hankel matrix by taking γ at random, for the probability of choosing a complex number γ for which C_γ was not diagonalizable would be null. However, an infinite number of values can lead to almost multiple eigenvalues, and for these values one can expect numerical difficulties. In the next section, we see that each coordinate of a Bézier curve of degree $2m-2$ is a Hankel form applied to the vector $\left(B_m^e(s)\right)^T e_m$ for each $s \in [0,1]$.

3. Bézier curve as a Hankel form. Bézier curves of degree $n-1$ ([5]) have become fundamental tools in Computed-Aided Geometric Design area. The de Casteljau's algorithm is a widespread method for the computation of these curves. But other efficient methods have arisen for this computation, e.g., those which evaluate polynomial curves expressed in generalized Ball bases ([12], [21], [24]), or use fast Pascal matrix-multiplication ([4]). In this section, we see that each coordinate of a Bézier curve can be represented by a Hankel form applied to a vector.

Let $Q_0 = (x_0, y_0)$, $Q_1 = (x_1, y_1)$, ..., $Q_{n-1} = (x_{n-1}, y_{n-1})$ be n points in $\mathbb{R}^2$. Assume that the number n is odd: $n = 2m-1$, $m > 1$. Then, for $k = 0, ..., m-1$,

$$B(s) = \sum_{j=0}^{k} \binom{k}{j} (1-s)^{k-j} s^j B_{Q_j Q_{j+1} \ldots Q_{j+n-k-1}}(s),$$

where $B_{Q_j Q_{j+1} \ldots Q_{j+n-k-1}}(s)$ denotes the Bézier curve defined by the points Q_j, Q_{j+1}, ..., $Q_{j+n-k-1}$ (the proof is left to the reader). Particularly for $k = m-1$,

$$B(s) = \sum_{j=0}^{m-1} \binom{m-1}{j} (1-s)^{m-1-j} s^j B_{Q_j Q_{j+1} \ldots Q_{j+m-1}}(s),$$

and so,

$$b_1(s) = \sum_{j=0}^{m-1} \binom{m-1}{j} (1-s)^{m-1-j} s^j e_m^T B_m^e(s) x_{j,\ldots,j+m-1},$$

$$b_2(s) = \sum_{j=0}^{m-1} \binom{m-1}{j} (1-s)^{m-1-j} s^j e_m^T B_m^e(s) y_{j,\ldots,j+m-1},$$

where $x_{j,\ldots,j+m-1}$ and $y_{j,\ldots,j+m-1}$ denote the column vectors $(x_j \ldots x_{j+m-1})^T$ and $(y_j \ldots y_{j+m-1})^T$, respectively, for $j = 0, ..., m-1$. Note that

$$\sum_{j=0}^{m-1} \binom{m-1}{j} (1-s)^{m-1-j} s^j e_m^T B_m^e(s) x_{j\ldots j+m-1} =$$

$$= e_m^T B_m^e(s) \left(\sum_{j=0}^{m-1} \binom{m-1}{j} (1-s)^{m-1-j} s^j x_{j\ldots j+m-1} \right).$$

Furthermore, $\sum_{j=0}^{m-1} \binom{m-1}{j} (1-s)^{m-1-j} s^j x_{j\ldots j+m-1}$ is a column vector whose first coordinate is $e_m^T B_m^e(s) \, (x_0 \ldots x_{m-1})^T$, the second coordinate is $e_m^T B_m^e(s) \, (x_1 \ldots x_m)^T$, and so on. Thus, we can state the following lemma:

LEMMA 3.1. *Let $n = 2m-1$, where m is an integer greater than 1. Let $B(s) = (b_1(s)\, b_2(s))^T$ be a Bézier curve of degree $n-1$ defined by n control points $Q_0 = (x_0, y_0)$, $Q_1 = (x_1, y_1)$, ..., $Q_{n-1} = (x_{n-1}, y_{n-1})$ in $\mathbb{R}^2$. Then*

$$b_1(s) = e_m^T B_m^e(s) H_x (B_m^e(s))^T e_m \text{ and } b_2(s) = e_m^T B_m^e(s) H_y (B_m^e(s))^T e_m,$$

where $H_x = hankel(C_x, R_x)$ and $H_y = hankel(C_y, R_y)$ are $m \times m$ Hankel matrices whose first columns are $C_x = (x_0 ... x_{m-1})^T$ and $C_y = (y_0 ... y_{m-1})^T$ respectively, and whose last rows are $R_x = (x_{m-1}, ..., x_{n-1})$ and $R_y = (y_{m-1}, ..., y_{n-1})$ respectively.

There is an analogous result for n even, as follows:

LEMMA 3.2. *Let $n = 2m$, where m is an integer greater than 1. Let $B(s) = (b_1(s)\, b_2(s))^T$ be a Bézier curve of degree $n-1$ defined by n points $Q_0 = (x_0, y_0)$, $Q_1 = (x_1, y_1)$, ..., $Q_{n-1} = (x_{n-1}, y_{n-1})$ in $\mathbb{R}^2$. Then*

$$b_1(s) = e_{m+1}^T B_m^e(s) H_x (B_m^e(s))^T e_m \text{ and } b_2(s) = e_{m+1}^T B_m^e(s) H_y (B_m^e(s))^T e_m,$$

where $H_x = hankel(C_x, R_x)$ and $H_y = hankel(C_y, R_y)$ are $(m+1) \times m$ Hankel matrices whose first columns are $C_x = (x_0 ... x_m)^T$ and $C_y = (y_0 ... y_m)^T$ respectively, and whose last rows are $R_x = (x_m, ..., x_{n-1})$ and $R_y = (y_m, ..., y_{n-1})$ respectively.

Proof. Let $H_x^0 = hankel(C_x^0, R_x^0)$ be an $m \times m$ Hankel matrix whose first column is $C_x^0 = (x_0 ... x_{m-1})^T$ and whose last row is $R_x^0 = (x_{m-1}, ..., x_{n-2})$. Let $H_x^1 = hankel(C_x^1, R_x^1)$ be an $m \times m$ Hankel matrix whose first column is $C_x^1 = (x_1 ... x_m)^T$ and whose last row is $R_x^1 = (x_m, ..., x_{n-1})$. Now

$$B(s) = B_{Q_0 Q_1 ... Q_{n-1}}(s) = (1-s) B_{Q_0 Q_1 ... Q_{n-2}}(s) + s B_{Q_1 Q_2 ... Q_{n-1}}(s).$$

Therefore,

$$b_1(s) = (1-s)[e_m^T B_m^e(s) H_x^0 (B_m^e(s))^T e_m] + s[e_m^T B_m^e(s) H_x^1 (B_m^e(s))^T e_m] =$$

$$= [(1-s)^m \binom{m-1}{1} (1-s)^{m-1} s \, ... \, \binom{m-1}{m-1} (1-s) s^{m-1} \, s^m] \begin{bmatrix} H_x^0 \\ 0 \end{bmatrix} (B_m^e(s))^T e_m +$$

$$+ [(1-s)^m \binom{m-1}{0} (1-s)^{m-1} s \, ... \, s^m] \begin{bmatrix} 0 \\ H_x^1 \end{bmatrix} (B_m^e(s))^T e_m =$$

$$= e_{m+1}^T B_{m+1}^e(s) \, diag([1, (m-1)/m, ..., (m-(m-1))/m, 1]) \begin{bmatrix} H_x^0 \\ 0 \end{bmatrix} (B_m^e(s))^T e_m +$$

$$+ e_{m+1}^T B_{m+1}^e(s) \, diag([1, 1/m, ..., (m-1)/m, 1]) \begin{bmatrix} 0 \\ H_x^1 \end{bmatrix} (B_m^e(s))^T e_m =$$

$$= e_{m+1}^T B_{m+1}^e(s) H_x (B_m^e(s))^T e_m.$$

Analogously, we would prove that $b_2(s) = e_{m+1}^T B_m^e(s) H_y (B_m^e(s))^T e_m$. ∎

The next theorem gives a new description of a Bézier curve and it is central to our paper.

THEOREM 3.3. *Let $n = 2m-1$, where m is an integer greater than 1. Let B be a Bézier curve of degree $n-1$ defined by n control points, and let $x = (x_0...x_{n-1})^T$ and $y = (y_0...y_{n-1})^T$ be their respective vector of coordinates. Let $H_x = hankel(C_x, R_x)$ and $H_y = hankel(C_y, R_y)$, where $C_x = (x_0...x_{m-1})^T$, $R_x = (x_{m-1}, ..., x_{n-1})$, $C_y = (y_0...y_{m-1})^T$ and $R_y = (y_{m-1}, ..., y_{n-1})$. If H_x and H_y are nonsingular, then there are complex numbers $d_1, ..., d_m$, $t_1, ..., t_m$, $\hat{d}_1, ..., \hat{d}_m$ and $\hat{t}_1, ..., \hat{t}_m$ such that*

$$b_1(s) = \sum_{i=1}^{m} d_i(1 - s + s.t_i)^{n-1} \text{ and } b_2(s) = \sum_{i=1}^{m} \hat{d}_i(1 - s + s.\hat{t}_i)^{n-1}. \tag{3.1}$$

Proof. Let $V = vander([t_1, ..., t_m])$ be a Vandermonde matrix and let $D = diag([d_1, ..., d_m])$ be a diagonal matrix such that $H_x = VDV^T$. So,

$$b_1(s) = e_m^T B_m^e(s) H_x (B_m^e(s))^T e_m = e_m^T B_m^e(s) V D V^T (B_m^e(s))^T e_m =$$

$$= \sum_{i=1}^{m} d_i(1 - s + s.t_i)^{2m-2} = \sum_{i=1}^{m} d_i(1 - s + s.t_i)^{n-1},$$

because $e_m^T B_m^e(s) V e_i = \sum_{j=0}^{m-1} (1-s)^{m-1-j}.s^j.t_i^j = (1 - s + s.t_i)^{m-1}$ for all $i \in \{1, ..., m\}$. In an analogous way, we conclude that

$$b_2(s) = \sum_{i=1}^{m} \hat{d}_i(1 - s + s.\hat{t}_i)^{n-1},$$

for some $\hat{d}_1, ..., \hat{d}_n$ and $\hat{t}_1, ..., \hat{t}_n$. □

Theorem 3.3 can be extended to any positive even integer n according to the following corollary, whose proof is an application of Lemma 3.2 and is left for the reader.

COROLLARY 3.4. *Let $n = 2m$, where m is an integer greater than 1. Given n control points $Q_0, ..., Q_{n-1}$, let $B_{Q_0Q_1...Q_{n-1}}(s) = (\, b_1(s)\, b_2(s)\,)^T$ be the Bézier curve of degree $n-1$ defined by these points. Let $x = (x_0...x_{n-1})^T$ and $y = (y_0...y_{n-1})^T$ be their respective vector of coordinates. Suppose that $H_x^0 = hankel(C_x^0, R_x^0)$ and $H_y^0 = hankel(C_y^0, R_y^0)$ are nonsingular, where $C_x^0 = (x_0...x_{m-1})^T$, $R_x^0 = (x_{m-1}, ..., x_{n-2})$, $C_y^0 = (y_0...y_{m-1})^T$ and $R_y^0 = (y_{m-1}, ..., y_{n-2})$. Then there are complex numbers $d_1, ..., d_m$, $t_1, ..., t_m$, $\hat{d}_1, ..., \hat{d}_m$ and $\hat{t}_1, ..., \hat{t}_m$ such that*

$$b_1(s) = \sum_{i=1}^{m} d_i(1 - s + s.t_i)^{n-1} \text{ and } b_2(s) = \sum_{i=1}^{m} \hat{d}_i(1 - s + s.\hat{t}_i)^{n-1}. \tag{3.2}$$

In the next section, we compute Bézier curves of degree $n-1$, $n = 2m-1$, from this Hankel form approach. We will compare the results obtained by our methods based on this approach with the ones obtained by de Casteljau's method and other methods. We will also discuss a preconditioning technique that we should use when the Hankel matrices H_x or H_y are ill-conditioned.

4. Numerical experiments. In this section we present results of several numerical experiments. We begin with the computation of Bézier curves which has been carried out by our methods and by four other methods. Since our methods depend on the good conditioning of H_x and H_y, we propose in 4.1 a procedure in order to improve the performance of our methods when at least one of these matrices is ill-conditioned.

Subdivisions and degree elevations of a Bézier curve are described from the point of view of Hankel forms in 4.2 and 4.3, respectively. In 4.4, we introduce a new way of computing a degree reduction of a Bézier curve. Finally, in 4.5 we briefly discuss the relationships between some basis obtained by the Hankel-matrix approach and corner-cutting systems.

We know that a uniform scaling of the control points of a Bézier curve yields a uniform scaling of the curve. We also note that if the control points are translated by a vector $v = (p, q)$, then the Bézier curve is also translated by v. Hence, without loss of generality, we are going to assume that the coordinates of the control points are all positive, and also less than or equal to 1. So, in order to generate n test control points we also use here the MATLAB function *rand*: $A = rand(n, 2)$. For the tests whose results are given in Table 4.1 we only considered sets of control points that yielded well-conditioned Hankel matrices H_x and H_y (both condition numbers less than 100).

Let $Q_0 = (x_0, y_0)$, $Q_1 = (x_1, y_1)$, ..., $Q_{n-1} = (x_{n-1}, y_{n-1})$ be n points of $\mathbb{R}^2$. Let X be the $n \times 2$ matrix defined by $X(i, 1) = x_{i-1}$ and $X(i, 2) = y_{i-1}$ for all $i \in \{1, ..., n\}$. Let $B = B(s)$ be the Bézier curve defined by these points. For each $i \in \{1, ..., 129\}$ let $s_i = (i - 1)/128$. Let C the 129×2 matrix such that $C(i, :)$ is the value of $B(s_i)$ computed by de Casteljau's algorithm for each $i \in \{1, ..., 129\}$. In an analogous way, let H, P, V, W and S be the 129×2 matrices so that, for all $i \in \{1, ..., 129\}$, $H(i, :)$, $P(i, :)$, $V(i, :)$, $W(i, :)$ and $S(i, :)$ are the values of $B(s_i)$ respectively computed by

1. our Algorithm 1;
2. a Pascal matrix algorithm that was introduced in [4];
3. the VS algorithm which was introduced in [22];
4. the Wang-Ball algorithm which was described in [24];
5. the Said-Ball algorithm, which was also described in [24].

The algorithms were implemented in MATLAB and are described as follows:

1. Our algorithms are divided into two parts: first, Vandermonde factorizations of the Hankel matrices H_x and H_y are computed; second, the expressions in 3.1 are evaluated. We will consider two algorithms in order to compute a Vandermonde factorization of a nonsingular Hankel matrix. The first algorithm, which is called Algorithm 1, solves an eigenvalue problem of the type $Cx = \lambda x$, where C is a companion matrix. Although Algorithm 1 can be implemented in $\mathcal{O}(m^2)$ operations, our version here demands $\mathcal{O}(m^3)$ operations. Our second algorithm, which is called Algorithm 2, solves a generalized eigenvalue problem of the type $H_0 v = \lambda H_1 v$, where H_0 and H_1 are Hankel matrices. Algorithm 2 demands $\mathcal{O}(m^3)$ operations. Both are explained as follows:
   In the first algorithm, although we could solve our two $m \times m$ Hankel systems in $\mathcal{O}(m^2)$ algebraic operations (see, e.g., [20]), we have preferred to use the MATLAB \ operation in our tests, which demanded $m^3/3 + m^2 - m/3$ multiplications and $m^3/3 + m^2/2 - 5m/6$ additions for each system. Since a companion matrix is already in Hessenberg form about $20/3\, m^3$ multiplications and additions are required in order to calculate the spectrum of each

Algorithm 1 Algorithm 1 - Vandermonde factorization of a regular Hankel matrix H_0 via companion matrix

- choose a number γ and define the vectors $x_\gamma = (h_{m+1} \ldots h_{2m-1}\, \gamma)^T$;
- solve the system $H_0 z_\gamma = x_\gamma$ and consider the companion matrix C_{z_γ};
- find the spectrum of C_{z_γ}: $\lambda(C_{z_\gamma}) = \{\alpha_1, ..., \alpha_m\}$;
- define $d = (V_\gamma)^{-1} He_1$, where $V_\gamma = vander([\alpha_1, ..., \alpha_m])$;

Algorithm 2 Algorithm 2 - Vandermonde factorization of a regular Hankel matrix H_0 via pencil

- choose a number γ and define the vectors $x_\gamma = (h_{m+1} \ldots h_{2m-1}\, \gamma)^T$;
- let $H_1 = [H_0(:, 2:m),\, x_\gamma]$;
- find the spectrum of $H_0 v = \lambda H_1 v$: $\{\alpha_1, ..., \alpha_m\}$;
- define $d = (V_\gamma)^{-1} He_1$, where $V_\gamma = vander([\alpha_1, ..., \alpha_m])$;

$m \times m$ companion matrix. Note that we needed $m(m-1)$ multiplications and $3/2m(m-1)$ additions to solve each Vandermonde system by using an algorithm described in [17]. After these calculations the evaluation of $B(s)$ demands $\mathcal{O}(m^2)$ operations for each $s \in (0,1)$. If we solve Hankel systems with an $\mathcal{O}(m^2)$-algorithm, and if the spectrum of the companion matrix is also calculated by an $\mathcal{O}(m^2)$-algorithm (see, e.g., [6], [11], [23]) the complexity of the whole process will be linear in m because the amount of operations demanded in the first step is divided by the number of values s in $(0,1)$. The second algorithm demands about $30\, m^3$ multiplications and additions to compute the same eigenvalues (see [17]).

2. The VS algorithm evaluates polynomials in the form $\sum_{i=0}^{n-1} P_i t^i (1-t)^{n-1-i}$ (see [14]). In order to compute points on the Bézier curve defined by the control points Q_0, ..., Q_{n-1}, we first pre-multiply the coordinates of each Q_i by $\binom{n-1}{i}$. Then we apply the VS algorithm, which consists of $2n-1$ products and $n-1$ additions for each $s \in (0,1)$.
3. The Wang-Ball algorithm evaluates polynomials in the Wang-Ball system (see [12]). Hence, we first change the coordinates of the initial control points by multiplying the vectors $X = (x_0 \ldots x_{n-1})^T$ and $Y = (y_0 \ldots y_{n-1})^T$ by an $n \times n$ matrix, which is not diagonal as in VS algorithm. Then we apply the Wang-Ball algorithm, which requires for each $s \in (0,1)$ $(3n-4)$ multiplications and $(3n-4)/2$ additions when $(n-1)$ is odd, and $3(n-1)$ multiplications and $3(n-1)/2$ additions when $(n-1)$ is even.
4. The Said-Ball algorithms evaluates polynomials in the Said-Ball system (see [14]). Like in the Wang-Ball and the VS algorithms, we first transform the coordinates of the control points by a change of basis matrix before applying the algorithm. Then, for each $s \in (0,1)$ the algorithm requires $n^2/2$ multiplications and $n^2/4$ additions for $(n-1)$ odd, and $(n-1)(\frac{n-1}{2}+2)$ multiplications and $\frac{n-1}{2}\frac{n-1}{2}+2)$ additions for $(n-1)$ even.
5. A Pascal matrix method computes a Bézier curve $B(s) = e_n^T B_n^e(s) X$ of degree $n-1$ via the decomposition $B_n^e(s) = P_n G_n(-s) P_n G_n(-1)$, where P_n is the $n \times n$ lower triangular Pascal matrix and $G_n(s) = diag([1, s, ..., s^{n-1}])$. The Pascal matrix method considered here consists of the following steps:
   - compute $Z = P_n G(-1) X$ and $W = P_n G(-1) Y$;
   - for each $s \in (0, 1/2]$ evaluate both the polynomials $e_n^T P_n G_n(-s) Z$ and

$e_n^T P_n G_n(-s)W$ by a Horner-like scheme;

- compute $Z_r = P_n G(-1) X_r$ and $W_r = P_n G(-1) Y_r$, where $X_r = X(n : -1 : 1)$ and $Y_r = Y(n : -1 : 1)$;
- for each $s \in (0, 1/2)$ evaluate both the polynomials $e_n^T P_n G_n(-s) Z_r$ and $e_n^T P_n G_n(-s) W_r$ by a Horner-like scheme;

Each Pascal matrix-vector multiplication consists of $n(n-1)/2$ additions. For each $s \in (0, 1/2)$ the algorithm requires $6(n-1)$ multiplications and $4(n-1)$ additions.

TABLE 4.1

Mean run time of computation in seconds of 129 points of a Bézier curve of degree N-1 by six different methods: our Algorithm 1 (H), VS (V), Wang-Ball (W), Said-Ball (S), a direct Pascal matrix method (P) and de Casteljau (C). NC means non-convergence.

N	Time (H)	Time (V)	Time (W)	Time (S)	Time (P)	Time (C)
31	0.0056	0.0007	0.10	0.65	0.0013	0.0093
39	0.0074	0.0008	0.13	1.03	0.0015	0.014
47	0.0088	0.0010	0.15	1.46	0.0018	0.019
55	0.011	0.0011	0.18	2.00	0.0020 (NC)	0.026
63	0.013	0.0012	0.22	2.68	0.0023 (NC)	0.034
71	0.015	0.0014	0.25	3.42	0.0026 (NC)	0.042
79	0.018	0.0015	0.28	4.23	0.0029 (NC)	0.052

In Table 4.1, we can compare the average computing times of the computation of Bézier curves of different degrees by several methods. Each result corresponds to the smallest time among 10 elapsed times obtained from consecutive executions of the respective procedure. They were computed by MATLAB's built-in *tic/toc* functions. For all methods the *tic* command is executed just after the definitions of the initial set of control points and of the mesh of time. This means that the amount of computing time includes the time demanded by the change of coordinates from Bernstein basis to another basis (Vandermonde, VS, Wang, Said and Pascal). Each NC in the column $Time(P)$ of the table means that $||P - C||_2 \geq 0.0005$. For instance, $||P - C||_2 = 0.065$ for $n = 55$.

The VS method had the best performance in time and accuracy. Our implementation of this method has used the command $abs(pascal(n, 1))$ instead of the command $nchoosek(n, k)$ in order to evaluate "n choose k". This simple procedure has substantially reduced the time of computation. We observe that this has occurred even after removing the warning message about the result being only accurate to 15 digits from the original code of the $nchoosek(n, k)$ function. For instance, the average computing time of the version of VS with such modified $nchoosek(n, k)$ function was 0.0050s for $n = 79$, while with $abs(pascal(n, 1))$ it was 0.0015s for the same value of n (see Table 4.1). In order to better understand the VS method recall how each coordinate of a Bézier curve $B(s)$ of degree $n - 1$ is described in matrix oriented language. For instance, $x(s)$ is defined by $e_n^T B_n^e(s) X$, where X is the vector of abscissas of the control points. Since $B_n^e(s) = G_n(1 - s) P_n G_n(s)/(1 - s)$, $x(s) = (1 - s)^{n-1} e_n^T P_n G_n(s/(1 - s)) X$. The VS method simply applies the Horner scheme to evaluate at $s/(1 - s)$ the polynomial whose coefficients are $\binom{n-1}{0} X_1$, ..., $\binom{n-1}{n-1} X_n$ for $s \leq 1/2$, and at $(1-s)/s$, the polynomial whose coefficients are $\binom{n-1}{0} X_n$, ..., $\binom{n-1}{n-1} X_1$ for $s > 1/2$. According to out experiments, this method has shown to be very accurate, at least up to $n = 79$. On the other hand, the Said-Ball method had the worst performance in time, although the number of arithmetic operations required by this method is almost the same of the one required by de Casteljau's method. This

bad performance in time was caused by the various conditional statements contained in the method, which demanded a precious time. However, this method is better suited to degree elevation than de Casteljau's method (see [24]), as well as the Pascal matrix method considered here. Nevertheless, the evaluation of the polynomials of degree $n-1$ in the Pascal method became unstable for $n \geq 55$ when s approached $1/2$: $||P-C||_2 = 3.8775$ for $n = 63$, $||P-C||_2 = 2.8077e+03$ for $n = 71$, and $||P-C||_2 = 5.2736e+05$ for $n = 79$. The evaluation is accurate up to $s = 1/4$. After this point, it becomes unstable.

With regard to Algorithm 1 that is based on Hankel forms, we can note that it had the third best performance in time. As for accuracy, after the VS method the Said-Ball method has obtained the best results, followed by our method and the Wang-Ball method. For this rank we have considered the results obtained by de Casteljau's method as the "true" values. However, as we have noticed before, when the two-condition number of H_x (or H_y) is large our methods become even more dependent on the number γ. In all the tests whose results were displayed in tables here, γ has been generated by the MATLAB function *rand*. From results obtained in MATLAB with a matrix returned by $A = rand(63, 2)$ (some of them are displayed in Table 4.2), we could totally verify that dependence. While we had been using *rand*, which generates numbers inside the interval $[0, 1]$, the results obtained by both Algorithms 1 and 2 were very far from the results obtained by de Casteljau's algorithm. However, for $\gamma = 1.3$, the norm of the difference between the set of 129 points computed by Algorithm 1 (2) and the one computed by de Casteljau's algorithm was 1.1963e-06 (8.2370e-05).

TABLE 4.2

Computation of 129 points of a Bézier curve of degree N-1 by three different methods: de Casteljau (C), Hankel form - Algorithm 1 (H1) and Hankel form - Algorithm 2 (H2). cond $= \max\{cond(H_x), cond(H_y)\}$*, where cond(A) is the two-condition number of the matrix A. The results of the second and third methods are compared to the ones obtained by de Casteljau's method via norm of the difference of the respective computed points. For each N there are three lines with the results obtained from three consecutive tests. For each test a random number is generated in order to expand an* $N \times N$ *Hankel matrix into an* $N \times (N+1)$*, which was used by Algorithm 1 and by Algorithm 2.*

N	cond	$\|\|C-H1\|\|_2$	$\|\|C-H2\|\|_2$
31	223.0	7.8110e-08	1.1214e-09
		6.8838e-07	2.1008e-08
		8.6408e-07	1.3360e-08
47	318.9	0.0033	4.4451e-04
		0.0019	5.1792e-05
		5.8218e-04	5.3150e-04
63	1323	7.9413e+52	1.9784e+52
		5.1264e+45	4.6812e+44
		3.3518e+36	3.0881e+34
79	198.4	0.0304	0.0057
		0.0731	7.9959e-04
		0.0340	0.0021

In order to overcome this problem we have included a procedure in our Algorithm 1 to improve the computation of points on the curves in those cases, which is explained in the following subsection.

4.1. Preconditioning the Hankel forms. It is not rare to find out that an $m \times m$ Hankel matrix H whose $2m-1$ entries are numbers taken at random in

TABLE 4.3

Mean run time of computation in seconds of 129 points of a Bézier curve of degree N-1 by three different methods: de Casteljau (C), Hankel form - Algorithm 1 (H) and preconditioning Hankel form (PH). cond $= \max\{cond(H_x), cond(H_y)\}$. *For each N the top line of* $||C - H||_2$, *or of* $||C - PH||_2$, *contains the minimum error among five consecutive tests; the bottom line, the maximum error. For each test a random number is generated in order to expand an* $N \times N$ *Hankel matrix into an* $N \times (N + 1)$.

N	cond	Time (C)	Time (H)	Time (PH)	$\|\|C - H\|\|_2$	$\|\|C - PH\|\|_2$
31	325.1	0.0094	0.0057	0.010	1.0447	3.5573e-13
					2.9059e+03	2.2654e-12
39	68.01	0.014	0.0074	0.013	2.4592e-11	1.7544e-12
					1.0558e-08	4.7451e-12
47	8.181e+04	0.020	0.0090	0.017	0.0098	4.9722e-12
					2.4884e+109	3.0472e-11
55	651.4	0.026	0.011	0.020	1.9364e-09	1.5026e-11
					2.0254e+20	2.9898e-11
63	548.1	0.034	0.013	0.022	1.1538	1.1737e-10
					2.5117e+07	3.5145e-10
71	198.8	0.042	0.015	0.027	2.1390e-09	7.7424e-11
					4.1055e-08	2.2024e-09
79	2.235e+03	0.052	0.018	0.031	9.5039e+21	7.4678e-09
					2.9080e+23	3.2787e-08

the interval [0,1] is an ill-conditioned matrix with respect to inversion. An attempt to handle the ill-conditioning of the matrix with respect to our methods is to shift its skew-diagonal in order to turn it into a skew-diagonally dominant matrix, $\tilde{H} = H + \sigma C$, where C is the reciprocal matrix ($C(:, k) = e_{m-k+1}$ for $k = 1, ..., m$). So, let B_H and $B_{\tilde{H}}$ be the Bézier curves corresponding to H and $\tilde{H}$ respectively. Then, for each $s \in [0, 1]$, we compute $B_H(s)$ by subtracting σ times $e_m^T B_m^e(s) C_m (B_m^e(s))^T e_m$ from $B_{\tilde{H}}$. Furthermore, this quadratic form has a simple formulation as can be seen in the next lemma.

LEMMA 4.1. *Let* $C_m = hankel(e_m, e_1^T)$, *which is called the reciprocal matrix. Then, if* $w = e^{2\pi i/m}$,

$$e_m^T B_m^e(s) C_m (B_m^e(s))^T e_m = \frac{1}{m} \sum_{j=1}^{m} w^{j-1} (1 - s + s.w^{j-1})^{n-1}.$$

Proof. It is easy to see that $C_m = VDV^T$, where $V = vander(1, w, ..., w^{m-1})$ and $D = diag(1/m, w/m, ..., w^{m-1}/m)$. From the proof of Theorem 3.3,

$$e_m^T B_m^e(s) V D V^T (B_m^e(s))^T e_m = \frac{1}{m} \sum_{j=1}^{m} w^{j-1} (1 - s + s.w^{j-1})^{n-1}.$$

□

In Table 4.3, we can see that preconditioning the Hankel forms associated to the matrices H_x and H_y, which is done by shifting their skew-diagonal, has improved the computation of Bézier curves. In the table $cond(H)$ is the maximum two-condition number between $cond(H_x)$ and $cond(H_y)$. For each Hankel matrix H, σ was taken as the sum of the absolute values of its entries. Since our computation of a Vandermonde factorization of a Hankel matrix depends on a value γ chosen at random, the

error between the curve computed by de Casteljau and the one computed from that factorization varies with γ, and the variation is used to be large when H_x and H_y are ill-conditioned. In table 4.3, for each n, we can see the minimum and the maximum errors among five consecutive experiments done. Notice that all our experiments have been run in an AMD Athlon 64 X2 Dual Core Processor 3600+ (2.00GHz), however, under a 32-bit MATLAB.

4.2. Subdivision. Let $B(s)$ a Bézier curve of degree $n-1$ ($n = 2m-1$) defined over $[0,1]$. Let $\hat{B}(s) = B(cs)$ be the part of the curve that corresponds to $[0,c]$, with $0 < c \leq 1$. Let $\hat{Q}_0 = B(0) = \hat{B}(0)$ and $\hat{Q}_{n-1} = B(c) = \hat{B}(1)$. Finding $n-2$ points, $\hat{Q}_1, ..., \hat{Q}_{n-2}$, such that $\hat{B}(s) = \hat{B}_{\hat{Q}_0\hat{Q}_1...\hat{Q}_{n-1}}(s)$ is referred to as subdivision of $B(s)$ ([15]). Suppose our method has calculated $B(s) = (b_1(s), b_2(s))$: $b_1(s) = \sum_{i=1}^{m} d_i^{(1)}(1-s+s.t_i^{(1)})^{n-1}$ and $b_2(s) = \sum_{i=1}^{m} d_i^{(2)}(1-s+s.t_i^{(2)})^{n-1}$. Let $\hat{V}_x = vander([\hat{t}_1^{(1)}, ..., \hat{t}_m^{(1)}])$, $D_x = diag([d_1^{(1)}, ..., d_m^{(1)}])$, $\hat{V}_y = vander([\hat{t}_1^{(2)}, ..., \hat{t}_m^{(2)}])$ and $D_y = diag([d_1^{(2)}, ..., d_m^{(2)}])$, where $\hat{t}_i^{(1)} = t_i^{(1)}.c - c + 1$ and $\hat{t}_i^{(2)} = t_i^{(2)}.c - c + 1$ for $i = 1, ..., m$. Therefore, it is not difficult to see that a new set of control points is given by (x_0, y_0), ..., (x_{n-1}, y_{n-1}), where $(x_0 \, ... \, x_{m-1})^T$ and $(x_{m-1} \, ... \, x_{n-1})$ are respectively the first column and the last row of $\hat{H}_x = \hat{V}_x D_x \hat{V}_x^T$, and $(y_0 \, ... \, y_{m-1})^T$ and $(y_{m-1} \, ... \, y_{n-1})$ are respectively the first column and the last row of $\hat{H}_y = \hat{V}_y D_y \hat{V}_y^T$.

4.3. Degree elevation. Applications that involve several Bézier curves can require that all these curves have the same degree. Increasing the degree of a Bézier curve without changing its shape is referred in the literature as degree elevation.

Let $n = 2m-1$ for some integer $m > 1$. Assume we have a Bézier curve of degree $n-1$ defined by n control points $Q_0, ..., Q_{n-1}$. Denote by H_x and H_y the $m \times m$ Hankel matrices associated to the respective coordinates of $B_{Q_0Q_1...Q_{n-1}}$. If we want to increase the degree of this curve to n without changing the shape we need a new set of $n+1$ control points Z_0, ..., Z_n. It is well known that these points can be computed as follows: $Z_0 = Q_0$ and

$$Z_k = \frac{k}{n} Q_{k-1} + \left(1 - \frac{k}{n}\right) Q_k, \quad 1 \leq k \leq n.$$

From matrix calculations we also arrive at the same result in terms of the Hankel matrices associated to the coordinates. Our approach turns out to be interesting for high degree elevation of the curve. In Figure 4.1 we can see the result of a degree elevation of a Bézier curve from 10 (its original degree) to 20 by our procedure, which will be explained in the following.

Suppose that we want to increase the degree of this curve to $N-1 = n-1+2r = 2(m+r-1)$, $r > 0$, without changing its shape. Let $\hat{H}_x$ and $\hat{H}_y$ be the $(m+r)\times(m+r)$ Hankel matrices associated to the coordinates of $B_{Z_0Z_1...Z_N}$. Now, for all $s \in [0,1]$,

$$e_{m+r}^T B_{m+r}^e(s) \hat{H}_x \left(B_{m+r}^e(s)\right)^T e_{m+r} =$$

$$= \sum_{i=0}^{m+r-1} \sum_{j=0}^{m+r-1} \binom{m+r-1}{i} \binom{m+r-1}{j} s^i s^j \hat{F}_{i+1,j+1},$$

where $\hat{F} = P_{m+r}^{-1} \hat{H}_x P_{m+r}^{-T} = hankel([\hat{f}_0; ...; \hat{f}_{m+r-1}], [\hat{f}_{m+r-1}, ..., \hat{f}_N])$. Observe that

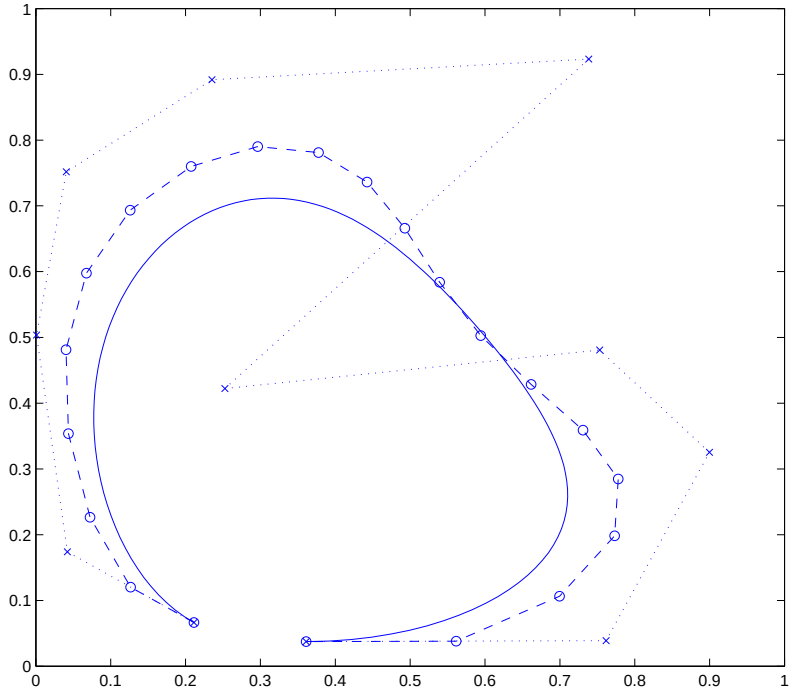

FIG. 4.1. *Degree elevation: Solid curve: Bézier curve of degree 10; dotted curve: initial set of 11 control points; dashed curve: new set of 21 control points*

$\hat{f}_k = 0$ if $k > n-1 = 2m-2$, for the degree of the curve is $n-1$. Now

$$e_{m+r}^T B_{m+r}^e(s)\hat{H}_x(B_{m+r}^e(s))^T e_{m+r} = \sum_{k=0}^{n-1}\left(\sum_{i+j=k}\binom{m+r-1}{i}\binom{m+r-1}{j}\right)\hat{f}_k s^k =$$

$$= \sum_{k=0}^{n-1}\binom{N-1}{k}\hat{f}_k s^k,$$

from Vandermonde convolution ([18]). Thus, for all $k \in \{0,...,n-1\}$ $\binom{N-1}{k}\hat{f}_k = \binom{n-1}{k}f_k$, where $F = P_m^{-1}H_x P_m^{-T}$. That is, we have $\hat{f}_0 = f_0$ and

$$(N-1)(N-2)...(N-k)\hat{f}_k = (n-1)(n-2)...(n-k)f_k, \text{ for all } 0 < k \leq n-1.$$

For instance, suppose $2r = n-1$, that is, $N = 2n-1$. Then,

$$\hat{F} = hankel([\hat{f}_0,...,\hat{f}_{n-1}],[\hat{f}_{n-1},0,....,0]),$$

where $\hat{f}_k = \dfrac{1}{2}\dfrac{1}{2+\dfrac{1}{n-2}}\cdots\dfrac{1}{2+\dfrac{k-1}{n-k}}f_k$ for $1 \leq k \leq n-1$. So, our algorithm to high degree elevation from degree $n-1$ to degree $2(n-1)$ becomes as follows:

- first we compute $H_x = VDV^T$. So, by defining $v_i = V(:,i)$ and $\alpha_i = D(i,i)$ for $i \in \{1,...,m\}$, we have

$$F = \sum_{i=1}^{m} d_i \left(P_m^{-1} v_i)\right) P_m^{-1} v_i\right)^T,$$

where $P_m^{-1} v_i = \left(1 \ (\alpha_i - 1) \ (\alpha_i - 1)^2 ... (\alpha_i - 1)^{m-1}\right)^T$. That is,

$$f_k = \sum_{i=0}^{m} d_i (\alpha_i - 1)^k.$$

- Second, we find $\hat{F}$ and compute a Vandermonde factorization of $\hat{F}$: $\hat{F} = W \Delta W^T$. Therefore, $\hat{H}_x = P_n \hat{F} P_n^T = (P_n W) \Delta (P_n W)^T$.
- Finally, by defining $w_i = We_i = \left(1 \ \beta_i ... \beta_i^{n-1}\right)^T$, we have that the abscissas of the new set of points are given by

$$\hat{h}_k = \sum_{i=1}^{n} \delta_i (\beta_i + 1)^k,$$

where $\delta_i = \Delta_{ii}$ for $1 \leq i \leq n$.

In the next subsection we discuss our method under degree reduction procedures.

4.4. Degree reduction. Decreasing the degree of a Bézier curve is a more complicated problem. In general we cannot reduce the degree of a curve without changing its shape, that is, degree reduction cannot be done exactly as degree elevation can. Let $B(s)$ be a Bézier curve of degree $n-1$, $n = 2m-1$. Suppose that the curve is described by the following equations as in 3.2:

$$b_1(s) = \sum_{i=1}^{m} d_i (1 - s + s.t_i)^{n-1} \text{ and } b_2(s) = \sum_{i=1}^{m} \hat{d}_i (1 - s + s.\hat{t}_i)^{n-1}.$$

Let r be a positive integer, $m < r < n-1$. Then the Bézier curve

$$b_1^{(m)}(s) = \sum_{i=1}^{m} d_i (1 - s + s.t_i^{\frac{n-1}{r}})^r \text{ and } b_2^{(m)}(s) = \sum_{i=1}^{m} \hat{d}_i (1 - s + s.\hat{t}_i^{\frac{n-1}{r}})^r$$

is a degree reduction of $B(s)$. So far our experiments have led us to consider this heuristic method in order to decrease the degree of a Bézier curve. In Figure 4.2 we can see this method applied to a Bézier curve of degree 14 which resulted in a Bézier curve of degree 11 very close to the original.

4.5. Corner cutting systems. Our method of computation of a Bézier curve $B(s)$ of degree $n-1$ ($n = 2m-1$) finds two sets of m numbers, $\{d_1,...,d_m\}$ and $\{\alpha_1,...,\alpha_m\}$, such that $B(s) = \sum_{k=1}^{m} d_k (\alpha_k s + 1)^{n-1}$ for each $s \in [0,1]$. Recall that if α_k and α_{k+1} are complex conjugates, then so are d_k and d_{k+1}.

Now suppose $\alpha_1,...,\alpha_n$ are n distinct numbers, which are either real or complex conjugate pairs. If α_k and α_{k+1} are complex conjugates, define $u_k = 2\,Re\,(\alpha_k s+1)^{n-1}$ and $u_{k+1} = -2\,Im\,(\alpha_k s+1)^{n-1}$; if α_k is real, let $u_k = (\alpha_k s+1)^{n-1}$. It is not difficult to verify that $\{u_1,...,u_n\}$ is a linearly independent set of functions on $[0,1]$. A sequence of real functions $\{u_1,...,u_n\}$ defined on $[0,1]$ will be called a system of functions.

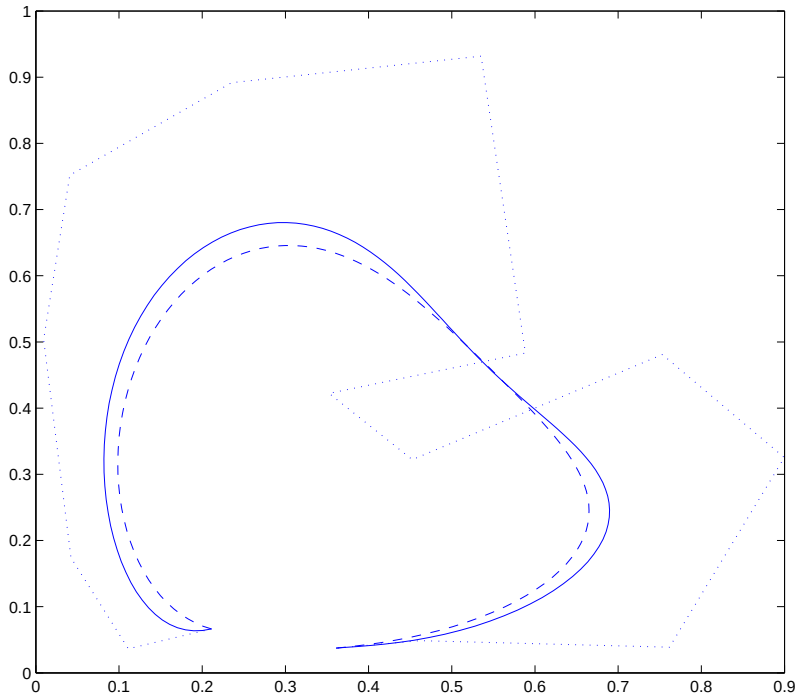

FIG. 4.2. *Degree reduction: Solid curve: Bézier curve of degree 14; dotted curve: initial set of 15 control points; dashed curve: Bézier curve of degree 11*

DEFINITION 4.2. *The collocation matrix of* $(u_1, ..., u_n)$ *at* $0 \leq s_1 < ... < s_k \leq 1$ *is given by a* $k \times n$ *matrix whose* $ij-$*entry is* $u_j(s_i)$ *for all* $1 \leq i \leq k$ *and* $1 \leq j \leq n$. *The system* $(u_1, ..., u_n)$ *is called* TP_2 *if all* 2×2 *minors of any collocation matrix of* $(u_1, ..., u_n)$ *are nonnegative.*

DEFINITION 4.3. *A system of functions* $(u_1, ..., u_n)$ *is monotonicity preserving if for any* $\beta_1 \leq \beta_2 \leq ... \leq \beta_n$ *in* $\mathbb{R}$, *the function* $\sum_{i=1}^{n} \beta_i u_i$ *is increasing.*

The following result is proved in [10].

PROPOSITION 4.4. *Let* $(u_1, ..., u_n)$ *be a system of functions defined on the interval* $[0, 1]$. *Let* $v_i = \sum_{j=i}^{n} u_j$ *for* $i \in \{1, ..., n\}$. *Then* $(u_1, ..., u_n)$ *is monotonicity preserving if and only if* v_1 *is a constant function and the functions* v_i *are increasing for* $i > 1$.

When $v_1 = 1$ we say that the system $(u_1, ..., u_n)$ is normalized. Moreover, if the functions $u_1, ..., u_n$ are nonnegative continuous functions, the system is called blending.

DEFINITION 4.5. *For each* $i \in \{1, ..., n-1\}$ *let* $\Delta_i(t)$ *be the* $i \times (i+1)$ *bidiagonal matrix such that* $(\Delta_i(t))_{kk} = 1 - \lambda_k^{(i)}(t)$ *and* $(\Delta_i(t))_{k,k+1} = \lambda_k^{(i)}(t)$ *for all* $k \in \{1, .., i\}$, *where* $\lambda_k^{(i)} : [0, 1] \to [0, 1]$ *is an increasing continuous function for each* $1 \leq k \leq i \leq n-1$. *We say that the* $1 \times n$ *matrix* $\Delta(t) = \Delta_1(t).\Delta_2(t). ....\Delta_{n-1}(t)$ *is a corner cutting representation on* $[0, 1]$ *of the system* $(u_1, ..., u_n)$ *if* $\Delta(t) = (u_1(t) \cdots u_n(t))$ *on* $[0, 1]$. *A system of functions admitting a corner cutting representation on* $[0, 1]$ *is called a*

corner cutting system on $[0,1]$.

A corner cutting system is an example of a blending system. Note that a corner cutting representation provides a corner cutting evaluation algorithm of a curve as

$$B(s)=\sum_{i=1}^{n}P_iu_i(t)=\Delta_1(t).\dots.\Delta_{n-1}(t)\begin{pmatrix}P_1\\ \vdots\\ P_n\end{pmatrix}.$$

One of the features of corner cutting systems is that they are monotonicity preserving. This fact and the following proposition are proved in [13].

PROPOSITION 4.6. *If* $(u_1,...,u_n)$ *is a blending* TP_2 *system of continuous functions on* $[0,1]$*, then it is a corner cutting system on* $[0,1]$.

From Proposition 4.6, an example of a corner cutting system of cubic polynomial functions on $[0,1]$ is $v_1=\frac{1}{34}[s^3-24s+32]$, $v_2=\frac{1}{17}[-2s^3+3s+1]$, $v_3=\frac{3}{170}[s^3-12s^2+24s]$ and $v_4=\frac{3}{85}[2s^3+6s^2+3s]$. These functions correspond to $2\,Re\,\beta_1.\,Re\,[\alpha_1s+1]^3$, $2\,Re\,\beta_2.\,Re\,[\alpha_2s+1]^3$, $-2\,Im\,\beta_1.\,Im\,[\alpha_1s+1]^3$ and $-2\,Im\,\beta_2.\,Im\,[\alpha_2s+1]^3$, respectively, where $\beta_1=\frac{8}{17}+i\frac{24}{85}$, $\beta_2=\frac{1}{34}-i\frac{3}{170}$, $\alpha_1=-\frac{1}{4}-i\frac{1}{4}$ and $\alpha_2=1+i$.

Let $Q_0=(x_0,y_0)$, $Q_1=(x_1,y_1)$, $Q_2=(x_2,y_2)$ and $Q_3=(x_3,y_3)$. Let $H_x=hankel([x_0,x_1,x_2],[x_2,x_3])$ and $H_y=hankel([y_0,y_1,y_2],[y_2,y_3])$. Suppose $V_1=vander([\alpha_1+1,\bar{\alpha}_1+1])$ and $V_2=vander([\alpha_2+1,\bar{\alpha}_2+1])$. Then

$$H_x=[V_1;(\alpha_1+1)^2,(\bar{\alpha}_1+1)^2]D_x^{(1)}V_1^T+[V_2;(\alpha_2+1)^2,(\bar{\alpha}_2+1)^2]D_x^{(2)}V_2^T,$$

$$H_y=[V_1;(\alpha_1+1)^2,(\bar{\alpha}_1+1)^2]D_y^{(1)}V_1^T+[V_2;(\alpha_2+1)^2,(\bar{\alpha}_2+1)^2]D_y^{(2)}V_2^T,$$

where $D_x^{(1)}=diag([a_1+b_1i,a_1-b_1i])$, $D_x^{(2)}=diag([c_1+d_1i,c_1-d_1i])$, $D_y^{(1)}=diag([a_2+b_2i,a_2-b_2i])$, $D_y^{(2)}=diag([c_2+d_2i,c_2-d_2i])$, $(a_1\,b_1\,c_1\,d_1)=(x_0\,x_1\,x_2\,x_3)M^T$, $(a_2\,b_2\,c_2\,d_2)=(y_0\,y_1\,y_2\,y_3)M^T$ and

$$M=\frac{1}{170}\begin{pmatrix}45&-128&120&-32\\ -10&19&-4&-8\\ 40&128&-120&32\\ -520&1056&-616&128\end{pmatrix}.$$

Hence, the Bézier curve defined by these points is

$$B(s)=\begin{pmatrix}a_1+b_1i\\ a_2+b_2i\end{pmatrix}(\alpha_1s+1)^3+\begin{pmatrix}a_1-b_1i\\ a_2-b_2i\end{pmatrix}(\bar{\alpha}_1s+1)^3+$$

$$+\begin{pmatrix}c_1+d_1i\\ c_2+d_2i\end{pmatrix}(\alpha_2s+1)^3+\begin{pmatrix}c_1-d_1i\\ c_2-d_2i\end{pmatrix}(\bar{\alpha}_2s+1)^3=$$

$$=\frac{1}{16}\begin{pmatrix}a_1\\ a_2\end{pmatrix}(s^3-24s+32)-\frac{1}{16}\begin{pmatrix}b_1\\ b_2\end{pmatrix}(s^3-12s^2+24s)+$$

$$+2\begin{pmatrix}c_1\\ c_2\end{pmatrix}(-2s^3+3s+1)-2\begin{pmatrix}d_1\\ d_2\end{pmatrix}(2s^3+6s^2+3s)=$$

$$= \frac{34}{16}\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} v_1 + 34\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} v_2 + (-\frac{170}{48})\begin{pmatrix} b_1 \\ b_2 \end{pmatrix} v_3 + (-\frac{170}{3})\begin{pmatrix} d_1 \\ d_2 \end{pmatrix} v_4.$$

Therefore, we see that for $n = 4$ we can obtain corner cutting systems of cubic functions on [0,1] by using pairs of conjugate complex functions like $\beta(\alpha s + 1)^n$ and $\bar{\beta}(\bar{\alpha}s+1)^n$. The obtention and use of such systems for all $n \geq 3$ are themes for future works.

5. Conclusion. Through this paper, we develop a matrix approach to the computation of Bézier curves from a set of n control points. We begin with our proof of the classical theorem of the existence of a Vandermonde factorization of a regular Hankel matrix, which is done by using Pascal matrix techniques. In the following, we introduced our main result: each coordinate of a Bézier curve of degree $n = 2m-1$ at $s \in [0, 1]$ can be represented by a Hankel form applied to the mth row of the $m \times m$ Bernstein matrix at s. From a Vandermonde factorization of these Hankel matrices, which are denoted by H_x and H_y, we see that we can easily compute the Bézier curve at s. However, the procedure for finding a Vandermonde factorization of a Hankel matrix depends on a value γ that is generated at random. Hence, if H_x or H_y is ill-conditioned, then the computation of the factors becomes even more sensitive to γ. We have dealt with this ill-conditioning of the problem by shifting the skew-diagonal of the ill-conditioned Hankel matrix to get a skew-diagonally dominant Hankel matrix. The effect of this shifting in the computation of the respective Bézier curve can be easily handled, according to Lemma 4.1. In Table 4.3 we can see some results of the application of this procedure. For more than one hundred matrices generated by $rand(n, 2)$ commands this procedure worked very well, and we could compute Bézier curves very close to the ones computed by de Casteljau's method. However, we have also seen that not only the condition numbers of H_x and H_y have to be small, but also a good choice of γ has to be made to guarantee a good accuracy. The numerical results have had a great variation depending on the value of γ. Strategies of choosing γ are a subject for a future work.

We also see that a subdivision formula for Bézier curves is easily obtained by our Hankel form approach. With respect to the behavior of our method under degree elevation, we have seen, as in Figure 4.1, that the method can be useful to high degree elevation. Otherwise, it is at least equivalent to de Casteljau's method. For degree reduction, we have proposed a new set of approximate Bézier curves of degree k, $m < k < n = 2m - 1$. In Figure4.2, we can observe a degree reduction from 14 to 11 according to our proposed method: the curves are very close one to the other. In §4.5 we have presented a corner cutting basis of $\mathbb{P}_{n-1}([0, 1])$, for $n = 4$, which is composed by functions of the type our method uses on the computation of Bézier curves. We have not yet obtained a general formula for a corner cutting system of $\mathbb{P}_{n-1}([0, 1])$ with these functions for all $n \geq 3$. A response for this problem is also left for a future work.

Summarizing, in our paper another proof of Vandermonde factorizations of Hankel matrices has been presented, a connection has been set up between a Bézier curve and a Hankel form, and from these results an efficient method of computation of a Bézier curve has been formulated. Also, we have introduced a subdivision formula for Bézier curves based on our approach and we have illustrated its behavior under degree reduction and degree elevation. Finally, we have introduced a corner cutting system on $[0, 1]$ for Bézier curves of degree 3 made up of functions of the type we have used in order to compute such curves by our approach.

REFERENCES

[1] L. Aceto and D. Trigiante, *The matrices of Pascal and other greats*, Amer. Math. Monthly, 108 (2001), pp. 232–245.

[2] A. A. Ball, *CONSURF, Part one: Introduction to conic lofting tile*, Comput. Aided Design, 6 (1974), pp. 243-249.

[3] B. Beckermann, G. H. Golub and G. Labahn, *On the numerical condition of a generalized Hankel eigenvalue problem*, Numer. Math., 106 (2007), pp. 41-68.

[4] L. H. Bezerra and L. K. Sacht, *On computing Bézier curves by Pascal matrix methods*, Appl. Math. Comput., 217 (2011), pp. 10118–10128.

[5] P. Bézier, *Numerical Control: Mathematics and Applications*, John Wiley & Sons, London, 1972.

[6] D. A. Bini, Y. Eidelman, L. Gemignani and I. Gohberg, *Fast QR Eigenvalue Algorithms for Hessenberg Matrices Which Are Rank-One Perturbations of Unitary Matrices*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 566-585.

[7] W. Boehm and A. Müller, *On de Casteljau's algorithm*, Comput. Aided Geom. D., 16 (1999), pp. 587–605.

[8] D. L. Boley, F. T. Luk and D. Vandervoorde, *A General Vandermonde Factorization of a Hankel Matrix*, in ILAS Symp. on Fast Algorithms for Control, Signals and Image Processing, 1997, Winnipeg.

[9] G. S. Call and D. J. Velleman, *Pascal's Matrices*, Amer. Math. Monthly, 100 (1993), pp. 372–376.

[10] J. M. Carnicer and J. M. Peña, *Totally positive bases for shape preserving curve design and optimality of B-splines*, Comput. Aided Geom. D., 11 (1994), pp. 633-654.

[11] S. Chandrasekaran, M. Gu, J. Xia and J. Zhu, *A Fast QR Algorithm for Companion Matrices*, Oper. Theory Adv. and Appl., 179 (2007), pp. 111-143.

[12] J. Delgado and J. M. Peña, *A shape preserving representation with an evaluation algorithm of linear complexity*, Comput. Aided Geom. D., 20 (2003), pp. 1–10.

[13] J. Delgado and J. M. Peña, *Corner cutting systems*, Comput. Aided Geom. Design, 22 (2005), pp. 81-97.

[14] J. Delgado and J. M. Peña, *On efficient algorithms for polynomial evaluation in CAGD*, Monogr. Semin. Mat. García de Galdeano, 31 (2004), pp. 111–120.

[15] G. E. Farin, *Curves and surfaces for computer aided geometric design: a practical guide*, Fourth ed., Academic Press, San Diego, CA, 1996.

[16] M. Fiedler, *Special Matrices and Their Applications in Numerical Mathematics*, Second ed., Dover, Mineola, NY, 2008.

[17] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Third ed., The Johns Hopkins University Press, Baltimore, MD, 1996.

[18] R. L. Graham, D. E. Knuth and O. Patashnik, *Concrete Mathematics - a Foundation for Computer Science*, Second ed., Addison-Wesley, Reading, MA, 1994.

[19] G. Heinig and K. Rost, *Algebraic methods for Toeplitz - like matrices and operators*, Birkhäuser, Basel, 1984.

[20] T. Kailath, *Displacement structure and array algorithms*, in Fast Reliable Algorithms for Matrices with Structure, T. Kailath and A. H. Sayed, eds., SIAM, Philadelphia, PA, 1999, pp. 1–56.

[21] H. N. Phien and N. Dejdumrong, *Efficient algorithms for Bézier curves*, Comput. Aided Geom. D., 17 (2000), pp. 247–250.

[22] L. L. Schumaker and W. Volk, *Efficient evaluation of multivariate polynomials*, Comput. Aided Geom. D., 3 (1986), pp. 149–154.

[23] M. Van Barel, R. Vandebril, P. Van Dooren, and K. Frederix, *Implicit double shift QR-algorithm for companion matrices*, Numer. Math., 116 (2010), pp. 177–212.

[24] H. Shi-Min, W. Guo-Zhao and J. Tong-Guang, *Properties of two types of generalized Ball curves*, Comput. Aided Design, 28 (1996), pp. 125–133.

[25] X. Wang and J. Zhou, *A fast eigenvalue algorithm for Pascal Matrices*, Appl. Math. Comput., 183 (2006), pp. 713–716.