Accurate computations with Lupaş matrices

Jorge Delgado [b] J. M. Peña [a]

[a] *Departamento de Matemática Aplicada, Universidad de Zaragoza, Spain*

[b] *Departamento de Matemática Aplicada, Universidad de Zaragoza, Spain*
Email address: jorgedel@unizar.es*; Phone number: +34978618174*

Abstract

Lupaş q-analogues of the Bernstein functions play an important role in Approximation Theory and Computer Aided Geometric Design. Their collocation matrices are called Lupaş matrices. In this paper, we provide algorithms for computing the bidiagonal decomposition of these matrices and their inverses to high relative accuracy. It is also shown that these algorithms can be used to perform to high relative accuracy several algebraic caculations with these matrices, such as the calculation of their inverses, their eigenvalues or their singular values. Numerical experiments are included.

Key words: accurate computations, bidiagonal decompositions, Lupaş operator, totally positive matrices

1 Introduction

The rapid development of q-calculus (see [12]), based on q-integers, has also influenced on the field of Approximation Theory. The Lupaş q-analogues of the Bernstein functions for $q > 0$ were introduced in [15]. Their applications can be seen in [16,19,20]. Recently (see[11]), they have been applied to Computer Aided Geometric Design (CAGD). In fact, Lupaş Bézier curves present many shape preserving properties (see [2]) because Lupaş matrices (the collocation matrices of the Lupaş q-analogues of the Bernstein functions) are totally positive (see Section 2). In this paper we shall prove that we can perform with Lupaş matrices many algebraic computations with high relative accuracy, including the computation of their inverses, their eigenvalues or their singular values. Up to now, such computations with high relative accuracy are possible

[1] Partially supported by MTM2015-65433 Spanish Research Grant and by Gobierno de Aragón.

with only a few classes of matrices, such as Vandermonde matrices and their generalizations [6], Bernstein-Vandermonde matrices [17], Said-Ball matrices [18], rational collocation matrices [3] or Jacobi-Stirling matrices [4].

Now let us introduce the basic definitions. Given a positive real number q we define a *q-integer* $[r]$ as

$$[r] = \begin{cases} 1 + q + \cdots + q^{r-1} = \frac{1-q^r}{1-q}, & \text{if } q \neq 1 \\ r, & \text{if } q = 1. \end{cases}$$

Then we define a *q-factorial* $[r]!$ as

$$[r]! = \begin{cases} [r] \cdot [r-1] \cdots [1], & \text{if } r \in \mathbb{N}, \\ 1, & \text{if } r = 0 \end{cases}$$

and finally we define the *q-binomial* coefficient as

$$\begin{bmatrix} n \\ r \end{bmatrix} = \frac{[n][n-1]\cdots[n-r+1]}{[r]!} = \frac{[n]!}{[r]![n-r]!}$$

for integers $n \geq r \geq 0$ and as zero otherwise. The *Lupaş q-analogues of the Bernstein functions* of degree n for $q > 0$ are defined as

$$l_{i,q}^n(t) = \frac{a_{i,q}^n(t)}{w_q^n(t)}, \quad t \in [0,1], \quad i = 0, 1, \ldots, n, \tag{1.1}$$

where

$$a_{i,q}^n(t) = \begin{bmatrix} n \\ i \end{bmatrix} q^{\frac{i(i-1)}{2}} t^i (1-t)^{n-i},$$
$$w_q^n(t) = \sum_{i=0}^{n} a_{i,q}^n(t) = \prod_{i=2}^{n} (1 - t + q^{i-1} t).$$

Let us observe that for the case $q = 1$ the Lupaş q-analogues of the Bernstein functions coincide with these polynomials, the most important polynomial basis used in CAGD.

The key tool to guarantee computations to high relative accuracy with Lupaş matrices is their bidiagonal decomposition, so that we can apply the algorithms of [14]. In Section 2, we obtain bidiagonal decompositions of Lupaş matrices and their inverses to high relative accuracy. In Section 3 we include numerical experiments illustrating the accuracy for computing the eigenvalues, singular values or the solution of linear systems with Lupaş matrices.

2 Bidiagonal decompositions of Lupaş matrices

We shall call the square collocation matrices $L_q := (l^n_{j,q}(t_i))_{0\le i,j\le n}$ of the Lupaş q-analogues of the Bernstein functions $(l^n_{0,q}, l^n_{1,q}, \ldots, l^n_{n,q})$ at a sequence of parameters $0 < t_0 < t_1 < \ldots < t_n < 1$ *Lupaş matrices.*

A matrix is *totally positive* (respectively, *strictly totally positive*) if all its minors are nonnegative (respectively, positive). These matrices have also been called in the literature as totally nonnegative and totally positive, respectively. There are many applications of these matrices in [7], [21] and [1]. From the results of [11], it can be derived that Lupaş matrices are totally positive. The following result shows that Lupaş matrices are in fact strictly totally positive.

Theorem 2.1 *Let $L_q = (l^n_{j,q}(t_i))_{0\le i,j\le n}$ be a Lupaş matrix whose nodes satisfy $0 < t_0 < t_1 < \ldots < t_n < 1$. Then:*

(a) $$\det \begin{pmatrix} (1-t_0)^n & t_0(1-t_0)^{n-1} & \cdots & t_0^n \\ (1-t_1)^n & t_1(1-t_1)^{n-1} & \cdots & t_1^n \\ \vdots & \vdots & \ddots & \vdots \\ (1-t_n)^n & t_n(1-t_n)^{n-1} & \cdots & t_n^n \end{pmatrix} = \prod_{0\le i<j\le n} (t_j - t_i).$$

(b) $$\det L_q = \frac{\prod_{k=0}^n \begin{bmatrix} n \\ k \end{bmatrix} q^{k(k-1)/2} \prod_{0\le i<j\le n}(t_j - t_i)}{\prod_{l=0}^n \prod_{k=2}^n (1 - t_l + q^{k-1} t_l)} (> 0).$$

(c) L_q is stritctly totally positive.

PROOF.

(a) It was proved in Theorem 3.1 (ii) of [3].

(b) All entries of the k-th column of the matrix L_q have as common factor $\begin{bmatrix} n \\ k-1 \end{bmatrix} q^{(k-1)(k-2)/2}$ and all the entries of the l-th row have as common factor $1/\prod_{k=2}^n(1 - t_{l-1} + q^{k-1}t_{l-1})$. Then, taking into account (a) of this theorem we have

$$\det L_q = \frac{\prod_{k=0}^n \begin{bmatrix} n \\ k \end{bmatrix} q^{k(k-1)/2}}{\prod_{l=0}^n \prod_{k=2}^n (1 - t_l + q^{k-1} t_l)} \prod_{0\le i<j\le n} (t_j - t_i).$$

(c) It is straightforward to check that each minor of a Lupaş matrix at parameters $0 < t_0 < t_1 < \ldots < t_n < 1$ has the same strict sign as the corresponding minor of the corresponding collocation matrix of the power basis at the positive parameters $t_0/(1-t_0) < t_1/(1-t_1) < \ldots < t_n/(1-t_n)$. This matrix is a Vandermonde matrix with positive and strictly increasing nodes, and so it is well known that it is strictly totally positive (cf. [7]), and the result follows.

Given $k \in \{1, 2, \ldots, n\}$ let $Q_{k,n}$ be the set of increasing sequences of k positive integers less than or equal to n. If $\alpha, \beta \in Q_{k,n}$, we denote by $A[\alpha|\beta]$ the $k \times k$ submatrix of A containing rows of places α and columns of places β. Besides let $A[\alpha] := A[\alpha|\alpha]$. Neville elimination is an elimination procedure to make zeros in a column of a matrix by subtracting to each row a multiple of the previous one (see [8]). It has played a key role for totally positive and strictly totally positive matrices (see [8], [9] and [10]). If $A = (a_{ij})_{0 \le i,j \le n}$ is a nonsingular matrix, this process has n steps, leading to a sequence of matrices: $A = A^{(0)} \to A^{(1)} \to \cdots \to A^{(n)} = U$, where U is an upper triangular matrix. The matrix $A^{(t+1)}$ is obtained from $A^{(t)}$ by the following formula

$$a_{ij}^{(t+1)} = \begin{cases} a_{ij}^{(t)}, & \text{if } 0 \le i \le j \le t, \\ a_{ij}^{(t)} - \dfrac{a_{it}^{(t)}}{a_{i-1,t}^{(t)}} a_{i-1,j}^{(t)}, & \text{if } t+1 \le i, j \le n \text{ and } a_{i-1,t}^{(t)} \ne 0, \\ a_{ij}^{(t)}, & \text{if } t+1 \le i \le n \text{ and } a_{i-1,t}^{(t)} = 0, \end{cases} \tag{2.1}$$

for $i = n, n-1, \ldots, t+1$ and for all $t \in \{0, 1, \ldots, n-1\}$.

The element

$$p_{ij} = a_{ij}^{(j)}, \quad 0 \le j \le i \le n, \tag{2.2}$$

is called the (i,j) *pivot* of Neville elimination of A. The pivots p_{ii} are called *diagonal pivots*. The Neville elimination can be performed without row exchanges if all the pivots are nonzero and, in this case, Lemma 2.6 of [8] implies that $p_{i0} = a_{i0}$ for $0 \le i \le n$ and

$$p_{ij} = \frac{\det A[i-j+1, \ldots, i+1|1, \ldots, j+1]}{\det A[i-j+1, \ldots, i|1, \ldots, j]} \tag{2.3}$$

for $0 < j \le i \le n$. The number

$$m_{ij} = \begin{cases} \dfrac{a_{ij}^{(j)}}{a_{i-1,j}^{(j)}} = \dfrac{p_{ij}}{p_{i-1,j}}, & \text{if } a_{i-1,j}^{(j)} \ne 0, \\ 0, & \text{if } a_{i-1,j}^{(j)} = 0, \end{cases} \tag{2.4}$$

is called the (i,j) *multiplier* of Neville elimination of A, where $0 \le j < i \le n$.

The next result can be derived from Theorem 4.1 of [8] and p. 116 of [10].

Theorem 2.2 *A matrix A is strictly totally positive if and only if the Neville elimination of A and A^T can be performed without row exchanges, all the mutipliers of the Neville elimination of A and A^T are positive and all the diagonal pivots of the Neville elimination of A are positive.*

We now present bidiagonal decompositions of Lupaş matrices by means of the diagonal pivots and multipliers of their Neville elimination and the multipliers of the Neville elimination of their transposes.

Theorem 2.3 *Let $L_q := (l^n_{j,q}(t_i))_{0\le i,j\le n}$ be a Lupaş matrix whose nodes satisfy $0 < t_0 < t_1 < \ldots < t_n < 1$. Then L_q can be written as*

$$L_q = F_n \cdots F_2 F_1 D G_1 G_2 \ldots G_n, \tag{2.5}$$

where F_i and G_i, $i \in \{1, \ldots, n\}$ are the lower and upper triangular bidiagonal matrices given by

$$F_i = \begin{pmatrix} 1 & & & & & & & \\ 0 & 1 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & 0 & 1 & & & & \\ & & & m_{i0} & 1 & & & \\ & & & & m_{i+1,1} & 1 & & \\ & & & & & \ddots & \ddots & \\ & & & & & & m_{n,n-i} & 1 \end{pmatrix}, \quad G_i^T = \begin{pmatrix} 1 & & & & & & & \\ 0 & 1 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & 0 & 1 & & & & \\ & & & \widetilde{m}_{i0} & 1 & & & \\ & & & & \widetilde{m}_{i+1,1} & 1 & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \widetilde{m}_{n,n-i} & 1 \end{pmatrix}, \tag{2.6}$$

and D the diagonal matrix $diag(p_{00}, p_{11} \ldots, p_{nn})$. The entries m_{ij}, $\widetilde{m}_{ij}$ and p_{ii} are given by

$$m_{ij} = \frac{(1-t_i)^{n-j}(1-t_{i-j-1})\prod_{k=2}^{n}(1-t_{i-1}+q^{k-1}t_{i-1})\prod_{k=i-j}^{i-1}(t_i-t_k)}{(1-t_{i-1})^{n+1-j}\prod_{k=2}^{n}(1-t_i+q^{k-1}t_i)\prod_{k=i-j-1}^{i-2}(t_{i-1}-t_k)}, \quad 0 \le j < i \le n,$$

$$\tilde{m}_{ij} = \frac{[n-i+1]q^{i-1}\,t_j}{[i](1-t_j)}, \quad 0 \le j < i \le n,$$

$$p_{ii} = \frac{\begin{bmatrix} n \\ i \end{bmatrix} q^{\frac{i(i-1)}{2}}(1-t_i)^{n-i}\prod_{k=0}^{i-1}(t_i-t_k)}{\prod_{k=2}^{n}(1-t_i+q^{k-1}t_i)\prod_{k=0}^{i-1}(1-t_k)}, \quad 0 \le i \le n.$$

PROOF. The matrix L_q is strictly totally positive by (c) of Theorem 2.1. So, by Theorem 2.2 the Neville elimination of L_q and its transpose L_q^T can be performed without row exchanges and, by Theorem 4.3 of [10] joint with the arguments of p. 116 of [10], we can obtain a bidiagonal factorization (2.5), where the nonzero off-diagonal entries of the factors F_i and G_i are the multipliers of the Neville elimination of L_q and L_q^T, respectively, and the diagonal entries of D are the diagonal pivots of the Neville elimination of L_q.

To obtain the pivots p_{ij}, by (2.3) we need to compute the minors of L_q with $j+1$ initial consecutive columns and $j+1$ consecutive rows starting from row $i-j+1$. Taking into account that all entries of the k-th column of the matrix $L_q[i-j+1,\ldots,i+1|1,\ldots,j+1]$ have as common factor $\begin{bmatrix} n \\ k-1 \end{bmatrix} q^{(k-1)(k-2)/2}$ and all the entries of the l-th row have as common factor $(1-t_{i-j+l-1})^{n-j}/\prod_{k=2}^{n}(1-$

$t_{i-j+l-1} + q^{k-1}t_{i-j+l-1})$, by (a) of Theorem 2.1, we have

$$\det L_p[i-j+1,\ldots,i+1|1,\ldots,j+1] = \frac{\prod_{k=0}^{j} \begin{bmatrix} n \\ k \end{bmatrix} q^{k(k-1)/2} \prod_{l=i-j}^{i}(1-t_l)^{n-j}}{\prod_{l=i-j}^{i} \prod_{k=2}^{n}(1-t_l+q^{k-1}t_l)} \prod_{i-j\le k<l\le i}(t_l-t_k).$$

By the previous formula and (2.3) we derive

$$\begin{aligned} p_{ij} &= \frac{\det L_p[i-j+1,\ldots,i+1|1,\ldots,j+1]}{\det L_p[i-j+1,\ldots,i|1,\ldots,j]} \\ &= \frac{q^{j(j-1)/2} \begin{bmatrix} n \\ j \end{bmatrix} (1-t_i)^{n-j} \prod_{k=i-j}^{i-1}(t_i-t_k)}{\prod_{k=2}^{n}(1-t_i+q^{k-1}t_i) \prod_{k=i-j}^{i-1}(1-t_k)}. \end{aligned}$$

Then, using (2.4), we conclude that

$$\begin{aligned} m_{ij} &= \frac{p_{ij}}{p_{i-1,j}} \\ &= \frac{(1-t_i)^{n-j}(1-t_{i-j-1}) \prod_{k=2}^{n}(1-t_{i-1}+q^{k-1}t_{i-1}) \prod_{k=i-j}^{i-1}(t_i-t_k)}{(1-t_{i-1})^{n+1-j} \prod_{k=2}^{n}(1-t_i+q^{k-1}t_i) \prod_{k=i-j-1}^{i-2}(t_{i-1}-t_k)}. \end{aligned}$$

Analogously, with the computation of the pivots $\tilde{p}_{ij}$ and the multipliers $\tilde{m}_{ij}$ of the Neville elimination of the matrix L_p^T, the result follows.

We shall denote the bidiagonal decomposition (2.5) and (2.6) of a Lupaş matrix L_q by $\mathcal{BD}(L_q)$. From $\mathcal{BD}(L_q)$, we can derive from Theorem 2.2 and the results in [8], [9] and [10] a bidiagonal decomposition of $(L_q)^{-1}$, given by

$$(L_q)^{-1} = \overline{G}_1\overline{G}_2\cdots\overline{G}_n D^{-1}\overline{F}_n\overline{F}_{n-1}\cdots\overline{F}_1, \tag{2.7}$$

where $\overline{F}_i$ and $\overline{G}_i$, $i \in \{1,\ldots,n\}$, are the lower and upper triangular bidiagonal matrices of the form of F_i and G_i (respectively), but replacing the off-diagonal entries $\{m_{i0}, m_{i+1,1},\ldots,m_{n,n-i}\}$ and $\{m_{i0}, m_{i+1,1},\ldots,m_{n,n-i}\}$ by the entries $\{-m_{i,i-1}, -m_{i+1,i-1},\ldots,-m_{n,i-1}\}$ and $\{-m_{i,i-1}, -m_{i+1,i-1},\ldots,-m_{n,i-1}\}$ (respectively).

Let us recall that an algorithm can be computed to high relative accuracy (HRA) when it only uses products, quotients, additions of numbers with the same sign or subtractions of initial data (cf. [5]). Since the explicit formulas for m_{ij}, p_{ii} and $\tilde{m}_{ij}$ in Theorem 2.3 only include substraction of initial data, $\mathcal{BD}(L_q)$ and the bidiagonal decomposition of $(L_q)^{-1}$ given by (2.7) can be calculated to HRA. These factorizations will be used later in order to solve certain linear systems $L_q x = b$, and to compute $(L_q)^{-1}$, and the eigenvalues and singular values of L_q in an accurate way.

A fast and accurate algorithm for computing the data corresponding to the bidiagonal factorization of the Lupaş matrix L_q and of its inverse has been developed by using the expressions of m_{ij}, $\tilde{m}_{ij}$ and p_{ii} in Theorem 2.3. Given the nodes $(t_i)_{0\le i\le n} \in (0,1)$, it returns a matrix $M \in \mathbb{R}^{(n+1)\times(n+1)}$ such that

$$\begin{aligned}
&M_{ii} = p_{i-1,i-1} \quad 1 \le i \le n+1,\\
&M_{ij} = m_{i-1,j-1} \quad 1 \le j < i \le n+1 \quad \text{and}\\
&M_{ij} = \tilde{m}_{j-1,i-1} \quad 1 \le i < j \le n+1.
\end{aligned}$$

Now we state in Algorithm 1 the procedure for the accurate computation of the multipliers m_{ij}. The algorithm is accurate because we only perform subtractions with the initial data.

Algorithm 1 Computation of the multipliers m_{ij}

Require: $(t_i)_{i=0}^n$ and $q > 0$
Ensure: m_{ij}
  for $i = 1$ to n **do**
    $M = \frac{(1-t_i)^n \prod_{k=2}^n \left[(1-t_{i-1})+q^{k-1}t_{i-1}\right]}{(1-t_{i-1})^{n+1}\prod_{k=2}^n\left[(1-t_i)+q^{k-1}t_i\right]}$
    $m_{i0} = (1-t_{i-1})M$
    for $j = 1$ to $i-1$ **do**
      $M = \frac{(1-t_{i-1})(t_i-t_{i-j})}{(1-t_i)(t_{i-1}-t_{i-j-1})}M$
      $m_{ij} = (1-t_{i-j-1})M$
    end for
  end for

Algorithm 2 provides the multipliers $\tilde{m}_{ij}$ with HRA.

Algorithm 2 Computation of the multipliers $\tilde{m}_{ij}$

Require: $(t_i)_{i=0}^n$ and $q > 0$
Ensure: $\tilde{m}_{ij}$
  for $j = 0$ to $n-1$ **do**
    $c_j = \frac{t_j}{1-t_j}$
    for $i = j+1$ to n **do**
      $\tilde{m}_{ij} = \frac{[n-i+1]}{[i]}q^{i-1}c_j$
    end for
  end for

The diagonal elements p_{ii} of D are computed accurately by Algorithm 3.

Clearly, the computation of algorithms 1-3 requires $\mathcal{O}(n^2)$ elementary operations.

Algorithm 3 Computation of the pivots p_{ii}

Require: $(t_i)_{i=0}^n$ and $q > 0$
Ensure: p_{ii}
  $r = 1$
  $p_{00} = \frac{(1-t_0)^n}{\prod_{k=2}^n \left[(1-t_0)+q^{k-1}t_0\right]}$
  for $i = 1$ to n **do**
    $r = \frac{[n+1-i]}{[i](1-t_{i-1})} r$
    $p_{ii} = r \, \frac{q^{i(i-1)/2}(1-t_i)^{n-i} \prod_{k=0}^{i-1}(t_i - t_k)}{\prod_{k=2}^n \left[(1-t_i)+q^{k-1}t_i\right]}$
  end for

3 Numerical experiments

In [13], assuming that the multipliers and diagonal pivots of an square totally positive matrix A and its transpose are known with HRA, Koev presents algorithms with HRA for computing the eigenvalues of A, its singular values and the solution of linear systems of equations $Ax = b$ where b has a chessboard pattern of alternating signs. In [14] we have a library, which contains an implementation of the three previous algorithms for using them with Matlab and Octave, and the name of the corresponding functions are `TNEigenvalues`, `TNSingularValues` and `TNSolve`, respectively. Their computational cost is $\mathcal{O}(n^2)$ elementary operations for `TNSolve` and $\mathcal{O}(n^3)$ for the first two functions. These three functions require as input argument the data determining the bidiagonal decompositions (2.5) of A and (2.7) of A^{-1}. `TNSolve` also requires a second argument, the vector b of the linear system $Ax = b$ to be solved. Observe that a computation of A^{-1} with HRA can be obtained directly by applying (2.7) with a cost of $\mathcal{O}(n^2)$ elementary operations.

In the previous section we have deduced how to compute the bidiagonal decomposition of Lupaş matrices and their inverses accurately (Algorithms 1, 2 and 3) with a total cost of $\mathcal{O}(n^2)$ elementary operations. We have implemented them in the function named `TNBDLupas` for Matlab and Octave, which take as input arguments a sequence of points $0 < t_0 < t_1 < \cdots < t_n < 1$ and a real number $q > 0$. The accurate bidiagonal decompositions of a Lupaş matrix obtained with `TNBDLupas` can be used with `TNEigenValues`, `TNSingularValues` and `TNSolve` in order to obtain accurate solutions for the above mentioned algebraic problems. Now we include some numerical experiments illustrating the high accuracy of the algorithms we have presented for solving the algebraic problems mentioned above.

First let us consider the Lupaş matrix A given by the collocation matrix of the Lupaş q-analogues of the Bernstein functions of degree 20 for $q = 0.5$

$(l^{20}_{0,0.5}, l^{20}_{1,0.5}, \ldots, l^{20}_{20,0.5})$ at the nodes $(t_i)_{0\le i\le 20}$, $t_i = (i+1)/22$, that is,

$$A = (l^{20}_{j,0.5}(t_i))_{0\le i,j\le 20}. \tag{3.1}$$

Now we consider the linear system $Ax = b$, where A is the Lupaş matrix previously defined and

$$\begin{aligned} b = (&35, -10, 8, -21, 95, -7, 13, -26, 83, -21, 64,\\ &-51, 88, -32, 27, -22, 7, -17, 11, -2, 11)^T \end{aligned} \tag{3.2}$$

has its entries randomly generated as integers in the interval $[1, 100]$ with a chessboard pattern of alternating signs. Then we obtain the exact solution of the linear system using Mathematica with exact arithmetic. We also compute two approximations $\widehat{x}$ to the solution x of the linear system with Matlab, the first one using `TNSolve` and the bidiagonal decomposition of the matrix A provided by `TNBDLupas`, and the second one using the Matlab command `A\b`. Finally, we calculate the corresponding componentwise relative errors for both approximations. In the case of the approximation obtained with command `A\b` the componentwise relative errors are always greater than 1.1, whereas, in the case of commands `TNBDLupas` and `TNSolve` the componentwise relative errors are always lower than $1.2 \cdot 10^{-15}$. Hence `TNSolve` with `TNBDLupas` provides a much more accurate approximation to the solution than the corresponding one to the approximation obtained with `A\b`. In fact, by using `TNSolve` with `TNBDLupas` $x = A^{-1}b$ is computed through the first formula of (2.7), and due to the signs of the factors of the bidiagonal decomposition and the signs of b, all computations have been performed with HRA.

We have also used the bidiagonal decomposition of the matrix A for computing the eigenvalues and the singular values of A with `TNEigenValues` and `TNSingularValues`, respectively. In the case of eigenvalues we also compute their approximations with the Matlab function `eig`. In order to determine the accuracy of the approximations to the eigenvalues computed in both ways we calculate the eigenvalues of the matrix A with Mathematica using a precision of 100 digits. Then we compute the relative errors corresponding to the approximations of the eigenvalues obtained with both methods `eig` and `TNEigenValues` wiht `TNBDLupas`, considering the eigenvalues provided by Mathematica as exact. The approximations of all the eigenvalues obtained with `TNBDLupas` are very accurate, whereas the approximations of the lower eigenvalues obtained with command `eig` are not very accurate. Table 1 shows the decimal form of six exact eigenvalues, including the four lowest, and the relative errors of the approximations to these eigenvalues (we order the eigenvalues of the greatest to the lowest $\lambda_1 > \lambda_2 > \cdots > \lambda_{21}$) obtained with both methods, one of them with HRA.

As for singular values, we have also computed their approximations with the

Matlab function `svd`. For determining the accuracy of the approximations to the singular values computed in both ways we calculate the singular values of the matrix A with Mathematica using a precision of 100 digits. Table 2 shows the four lowest singular values computed with Mathematica and, the relative errors of the singular values computed in Matlab with `svd`, and with `TNBDLupas` and `TNSingularValues`, considering the singular values provided by Mathematica as exact (we order the singular values of the greatest to the lowest $\sigma_1 > \sigma_2 > \cdots > \sigma_{21}$).

On the one hand, we can observe that all the eigenvalues and singular values computed using the bidiagonal decomposition of the Lupaş matrix are very accurate. On the other hand, not all the eigenvalues and singular values computed with `eig` and `svd` Matlab functions, respectively, are accurate. In particular, the lower the eigenvalue (resp., singular value) is, the more unaccurate the approximation obtained with `eig` (resp., `svd`) is. In order to illustrate this fact more generally, we have computed with Matlab the lowest eigenvalue and the lowest singular value of the Lupaş matrices of order $5, 7, 9, \ldots, 41$ with nodes $(t_i)_{i=0}^n$ given by

$$t_i = \frac{i+1}{n+2}, \quad i = 0, 1, \ldots, n, \tag{3.3}$$

for the values $n = 4, 6, 8, \ldots, 40$, with `eig` and `svd` functions, and with `TNBDLupas`, `TNEigenValues` and `TNSingularValues` functions. Figure 1 shows the relative errors of the results obtained for the different considered orders. Once again, `TNBDLupas` with `TNEigenValues` and `TNSingularValues` outperforms `eig` and `svd`.

References

[1] T. Ando, Totally positive matrices, Linear Algebra Appl. 90 (1987) 165–219.

[2] J. M. Carnicer, J. M. Peña, Shape preserving representations and optimality of the Bernstein basis, Adv. Comput. Math. 1 (1993) 173–196.

[3] J. Delgado, J. M. Peña, Accurate computations with collocation matrices of rational bases, Appl. Math. Comput. 219 (2013) 4354–4364.

[4] J. Delgado, J. M. Peña, Fast and accurate algorithms for Jacobi-Stirling matrices, Applied Mathematics and Computation 236 (2014) 253–259.

[5] J. Demmel, M. Gu, S. Eisenstat, I. Slapnicar, K. Veselic and Z. Drmac, Computing the singular value decomposition with high relative accuracy, Linear Algebra Appl. 299 (1999) 21–80.

[6] Demmel J, Koev P. The accurate and efficient solution of a totally positive generalized Vandermonde linear system, SIAM J. Matrix Anal. Appl. 27 (2005) 142–152.

[7] S. M. Fallat, C. R. Johnson, Totally nonnegative matrices, Princeton Series in Applied Mathematics, Princeton University Press, Princeton, NJ, 2011.

[8] M. Gasca, J.M. Peña, Total positivity and Neville Elimination, Linear Algebra Appl. 165 (1992) 25–44.

[9] M. Gasca, J.M. Peña, A matricial description of Neville elimination with applications to total positivity, Linear Algebra Appl. 202 (1994) 33–53.

[10] M. Gasca, J.M. Peña, On factorizations of totally positive matrices. In: *Total Positivity and Its Applications* (M. Gasca and C.A. Micchelli, Ed.), Kluver Academic Publishers, Dordrecht, The Netherlands (1996), 109–130.

[11] L.W. Han, Y. Chu, Z.Y. Qiu, Generalized Bézier curves and surfaces based on Lupaş q-analogue of Bernstein operator, Journal of Computational and Applied Mathematics 261 (2014) 352–363.

[12] V. Kac, P. Cheung, Quantum calculus, Universitext, Springer-Verlag, New York, 2002.

[13] P. Koev, Accurate Computations with totally nonnegative matrices, SIAM J. Matrix Anal. Appl. 29 (2007) 731–751.

[14] P. Koev, `http://math.mit.edu/~plamen/software/TNTool.html`

[15] A. Lupaş, A q-analogue of the Bernstein operator, Seminar on Numerical and Statistical Calculus, 9 (1987) 85–92.

[16] N.I. Mahmudov, P. Sabancigil, Voronovskaja type theorem for the Lupaş q-analogue of the Bernstein operators, Mathematical Communications 17 (2012) 83–91.

[17] A. Marco, J.-J. Martínez, A fast and accurate algorithm for solving Bernstein-Vandermonde linear systems, Linear Algebra Appl. 422 (2007) 616–628.

[18] A. Marco, J.-J. Martínez, Accurate computations with Said-Ball-Vandermonde matrices, Linear Algebra Appl. 432 (2010) 2894–2908.

[19] S. Ostrovska, On the Lupaş q-analogue of the Bernstein operator, Journal of Mathematics 36 (2006) 1615–1629.

[20] S. Ostrovska, On the Lupaş q-transform, Computers and Mathematics with Applications 61(2011) 527–532.

[21] A. Pinkus, Totally positive matrices, Cambridge Tracts in Mathematics, 181. Cambridge University Press, Cambridge, 2010.

i	λ_i	rel.errors with HRA	rel.errors with `eig`
1	$1.0000e+00$	$0.0000e+00$	$2.2204e-16$
2	$9.5727e-02$	$1.4497e-16$	$1.5947e-15$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
18	$2.3984e-34$	$5.7057e-15$	$1.5025e-09$
19	$5.7754e-36$	$1.1107e-14$	$6.0675e-10$
20	$1.0913e-36$	$7.1947e-15$	$1.4395e-09$
21	$2.2562e-38$	$1.6543e-14$	$2.3703e-09$

Table 1
Eigenvalues of a Lupaş matrix of order 21

i	σ_i	rel.errors with HRA	rel.errors with `svd`
1	$2.2869e+00$	$0.0000e+00$	$1.9419e-16$
2	$1.3443e+00$	$6.6068e-16$	$1.6517e-16$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
18	$6.9344e-41$	$3.9699e-15$	$4.6494e+01$
19	$1.0242e-46$	$3.2278e-15$	$4.8234e+03$
20	$4.9347e-53$	$6.0132e-15$	$4.4492e+04$
21	$5.2446e-60$	$4.4262e-15$	$7.2861e+06$

Table 2
Singular values of a Lupaş matrix of order 21

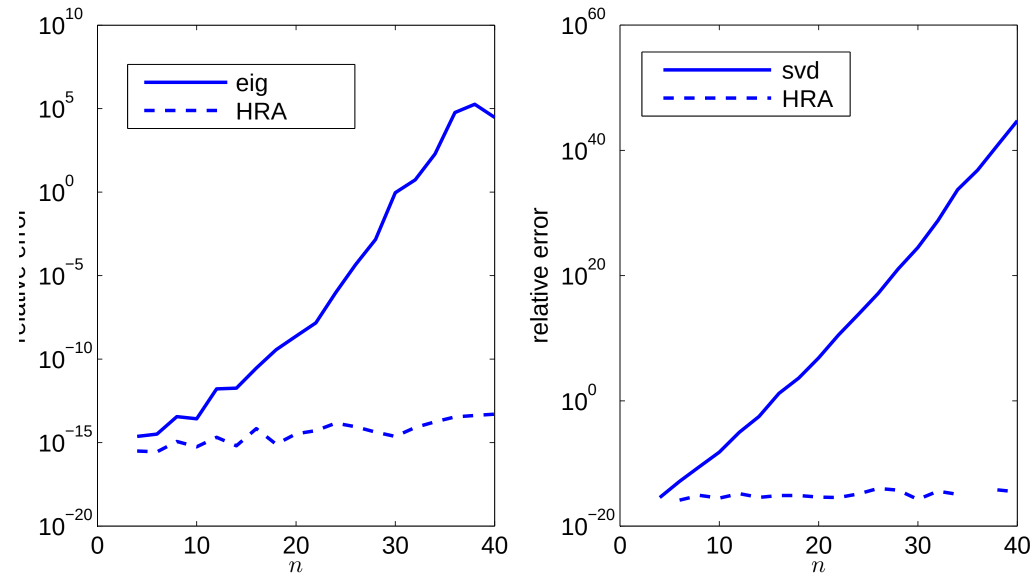

Fig. 1. Relative errors for the lowest eigenvalue and the lowest singular value of Lupaş matrices