Fast Approximation for Toeplitz, Tridiagonal, Symmetric and Positive Definite Linear Systems that Grow Over Time

Pedro Mayorga*, Alfonso Estudillo, A. Medina-Santiago, José Vázquez and Fernando Ramos
Faculty of Biomedical Engineering
Universidad Polit´ecnica de Chiapas
Chiapas, M´exico

***Abstract*—Linear systems with tridiagonal structures are very common in problems related not only to engineering, but chemistry, biomedical or finance, for example, real time cubic B-Spline interpolation of ND-images, real time processing of Electrocardiography (ECG) and hand drawing recognition. In those problems which the matrix is positive definite, it is possible to optimize the solution in $\mathcal{O}(n)$ time. This paper describes such systems whose size grows over time and proposes an approximation in $\mathcal{O}(1)$ time of such systems based on a series of previous approximations. In addition, it is described the development of the method and is proved that the proposed solution converges linearly to the optimal. A real-time cubic B-Spline interpolation of an ECG is computed with this proposal, for this application the proposed method shows a global relative error near to 10^{-6} and its computation is faster than traditional methods, as shown in the experiments.**

***Keywords*—real time interpolation; linear convergence; Cholesky decomposition; biomedical data acquisition**

I. Introduction

Several problems in many fields of science are related to linear systems of the form $Ax = b$ [1], [2], [3]. Matrices with special structures such as the Toeplitz matrix arise in many problems, including the solution of ordinary and partial differential equations [4], [5], [6], [2], [7], [8]. In many cases the size of these systems is very large; then, computing the solution in a reasonable amount of time becomes a problem for real time applications [9], [10], [11], [12], [13], [14], [3], [15]. Due to the increasing interest in tridiagonal matrices, it is important to understand its properties and its applications [16], [12], [17], [18], [19], [20], [21], [22], [23], [24].

In general, the system $Ax = b$ is solved in $\mathcal{O}(n^2)$ time [25]; if A is tridiagonal, i.e. $A_{ij} = 0$ for $|i - j| > 1$, there exists fast algorithms to solve it in $\mathcal{O}(n)$ time [16], [11], [2], [19], [3], [25], [24]. Additionally, if A is Toeplitz, symmetric and positive definite as shown in (1), the computation can be optimized [16], [11], [2], [3], for example by using the Cholesky decomposition $A = LL^T$ [11], [3]. Equation (1) describes all of those matrices with constant diagonal and (2) represents the ratio of the matrix coefficients that must lie between -1 and 1.

$$A = \begin{bmatrix} \beta & \alpha & & & \\ \alpha & \beta & \alpha & & \\ & \ddots & \ddots & \ddots & \\ & & \alpha & \beta & \alpha \\ & & & \alpha & \beta \end{bmatrix}, \qquad \beta > 2\,|\alpha|\,. \tag{1}$$

$$-1 < \nu \equiv \frac{2\alpha}{\beta} < 1. \tag{2}$$

A. Cholesky decomposition

For a square, symmetric and positive definite matrix A, the Cholesky decomposition computes a matrix L such that $LL^T = A$. The coefficients of L can be computed in $\mathcal{O}(n)$ [3] following the iterative scheme (3):

$$\begin{aligned} L_{11}^2 &= A_{11}, \\ L_{i,i-1} &= A_{i,i-1}/L_{i-1,i-1}, \\ L_{ii}^2 &= A_{ii} - L_{i,i-1}^2, \quad \text{for} \quad i = 2, \ldots, n. \end{aligned} \tag{3}$$

When the matrix A has the structure defined in (1), the coefficients can be computed in a closed form [16], [19], as shown in (4). Those coefficients have the following limits: $\lim_{i\to\infty} L_{ii} = \sqrt{\lambda_1}$ and $\lim_{i\to\infty} L_{i,i-1} = \alpha/\sqrt{\lambda_1}$, due to the fact that $\beta > 2|\alpha|$ and $\lambda_1 > \lambda_2 > 0$.

$$\begin{aligned} L_{ii} &= \sqrt{\tfrac{\lambda_1^{i+1} - \lambda_2^{i+1}}{\lambda_1^i - \lambda_2^i}}, \\ L_{i,i-1} &= \alpha\sqrt{\tfrac{\lambda_1^i - \lambda_2^i}{\lambda_1^{i+1} - \lambda_2^{i+1}}}, \quad \text{where} \\ \lambda_1 &= \tfrac{\beta + \sqrt{\beta^2 - 4\alpha^2}}{2},\ \lambda_2 = \tfrac{\beta - \sqrt{\beta^2 - 4\alpha^2}}{2}. \end{aligned} \tag{4}$$

With this approach the system can be computed in $\mathcal{O}(n)$ and is able to be parallelized [9], [10], [14].

B. Solve $Ax = b$ using Cholesky decomposition

The Cholesky decomposition allows to find the solution x of the linear system $Ax = b$ by first finding u such that $Lu = b$ and, then finding x such that $L^T x = u$. If A is defined as in (1), the Cholesky decomposition can be used to solve the linear system in $\mathcal{O}(n)$ time. The complete iterative procedure is described as follows:

$$\begin{aligned} u_1 &= \frac{b_1}{L_{11}}, \\ u_i &= \frac{b_i - L_{i,i-1}u_{i-1}}{L_{ii}}, \quad \text{for } i = 2,3,\ldots,n; \end{aligned} \tag{5}$$

after computing all u_i, the x_i is computed as:

$$\begin{aligned} x_n &= \frac{u_n}{L_{nn}}, \\ x_i &= \frac{u_i - L_{i+1,i}x_{i+1}}{L_{ii}}, \quad \text{for } i = n-1,\ldots,1. \end{aligned} \tag{6}$$

II. Material and Methods

This section shows the computation of an approximation of the $(n+1)\times(n+1)$ linear system, based on the solution of a similar system of size $n \times n$.

A. A linear system of size $(n+1)\times(n+1)$

Let $Ax = b$ be a linear system, where $A \in \mathbb{R}^{n\times n}$ with structure defined in (1), and $x, b \in \mathbb{R}^n$.

The solution of this kind of systems can be computed with the procedure described in (5) and (6) in $\mathcal{O}(n)$ time. Now, consider an expanded version of the system $Ax = b$ as $A^+x^+ = b^+$, where $A^+ \in \mathbb{R}^{(n+1)\times(n+1)}$ and $x^+, b^+ \in \mathbb{R}^{n+1}$, such that, A^+ has the same structure as (1) and $b_j^+ = b_j$, for $j = 1,2,\ldots,n$.

The system $A^+x^+ = b^+$ appears in problems that grow over time, such as: real time interpolation of biomedical data (ND Images or ECG), filtering data, handwriting recognition and real time image processing. Solving $A^+x^+ = b^+$ requires additional $\mathcal{O}(n+1)$ time, therefore, solving all systems from size 1 to n is $\mathcal{O}(n^2)$, i.e. this approach is not convenient.

This paper proposes a new method that computes an approximation of $A^+x^+ = b^+$, based on the solution of $Ax = b$ in $\mathcal{O}(1)$ time, moreover, it is demonstrated that the proposed method has linear convergence and it requires only 6 steps to reach a relative error about 10^{-4}. The method is summarized in algorithm 1.

Algorithm 1 Approximation of $A^+x^+ = b^+$ based on $Ax = b$ in $\mathcal{O}(1)$ time

```
Require: $Ax = b$ of size $n$ with computed solution $x$.
Require: $A^+x^+ = b^+$ of size $n+1$ s.t. $b_i^+ = b_i$ for $i = 1,\ldots,n$.
Require: an small integer $k > 0$                    ▷ typically $k = 6$
1: Set $A_{k+1}$ the matrix (1) of size $k+1$
2: Set $r_1 = b_{n-k+1}^+ - \alpha x_{n-k}$           ▷ update last value
3: Set $r_i = b_{n-k+i}^+$ for $i = 2,\ldots,k+1$
4: Solve $A_{k+1}u = r$
5: Set $x_i^+ = x_i$ for $i = 1,\ldots,n-k$
6: Set $x_{n-k+i}^+ = u_i$ for $i = 1,\ldots,k+1$
```

The next paragraphs expose the basis of the method as follows:

- Cholesky decomposition used to solve $Ax = b$ and $A^+x^+ = b^+$ shows that intermediate solution u and u^+ share the same values (theorem 1);
- theorem 2 and corollary 1 show that the error between x_i^+ and x_i increases from $i = 1$ to n or decreases from $i = n$ to 1;
- fig. 5 shows that $|x_i^+ - x_i| \propto 10^{-4}$ for $i = 1,\ldots,n-6$. The last point suggests that only is necessary to solve a little system to get an approximation of x^+ as shown in algorithm 1.

Theorem 1. *The first n terms of the partial solution u^+ are identical to the first n terms of the partial solution u, i.e.,*

$$u_j^+ = u_j, \quad \textit{for } j = 1,2,\ldots,n. \tag{7}$$

Proof: Since A^+ has the same structure as A, then $L_{ij}^+ = L_{ij}$ for $i,j = 1,\ldots,n$; and using $b_i^+ = b_i$ for $i = 1,\ldots,n$:

$$u_1^+ = \frac{b_1^+}{L_{11}^+} = \frac{b_1}{L_{11}} = u_1,$$

and by induction

$$u_i^+ = \frac{b_i^+ - L_{i,i-1}^+u_{i-1}^+}{L_{ii}^+} = \frac{b_i - L_{i,i-1}u_{i-1}}{L_{ii}} = u_i, \quad \text{for } i = 2,3,\ldots,n.$$

■

Theorem 2. *The absolute error between x_i^+ and x_i is an increasing function of i, moreover*

$$x_i^+ - x_i = (-1)^{n-i+1}\left(\prod_{k=i}^{n}\frac{L_{k+1,k}}{L_{kk}}\right)x_{n+1}^+, \quad i \le n. \tag{8}$$

Proof: First, the proof of equation (8) is given; second, the proof of equation (8) is shown.

The principle of induction is used to prove (8) as follows: 1) equation (8) is true for $i = n$ (use equation (6)):

$$\begin{aligned} x_n^+ - x_n &= \left(\frac{u_n^+}{L_{nn}^+} - \frac{L_{n+1,n}^+}{L_{nn}^+}x_{n+1}^+\right) - \left(\frac{u_n}{L_{nn}}\right) \\ &= \left(\frac{u_n}{L_{nn}} - \frac{L_{n+1,n}}{L_{nn}}x_{n+1}^+\right) - \left(\frac{u_n}{L_{nn}}\right) \\ &= -\frac{L_{n+1,n}}{L_{nn}}x_{n+1}^+; \end{aligned}$$

$\therefore$ (8) is true for $i = n$; 2) assume for a moment that (8) is true for $i = n-j+1$, given $x_{n-j+1}^+ - x_{n-j+1} = (-1)^j\left(\prod_{k=n-j+1}^{n}\frac{L_{k+1,k}}{L_{kk}}\right)x_{n+1}^+$; 3) the following lines show the truth for $i = n - j$:

$$\begin{aligned}
&x^+_{n-j} - x_{n-j} \\
&= \frac{u_{n-j} - L_{n-j+1,n-j}x^+_{n-j+1}}{L_{n-j,n-j}} \\
&\quad - \frac{u_{n-j} - L_{n-j+1,n-j}x_{n-j+1}}{L_{n-j,n-j}} \\
&= -\frac{L_{n-j+1,n-j}}{L_{n-j,n-j}}\left(x^+_{n-j+1} - x_{n-j+1}\right) \\
&= -\frac{L_{n-j+1,n-j}}{L_{n-j,n-j}}\left((-1)^j\left(\prod_{k=n-j+1}^{n}\frac{L_{k+1,k}}{L_{k,k}}\right)x^+_{n+1}\right) \\
&= (-1)^{j+1}\left(\prod_{k=n-j}^{n}\frac{L_{k+1,k}}{L_{k,k}}\right)x^+_{n+1}.
\end{aligned}$$

$\therefore$ (8) is true for $i = n - j$.

To prove that $|x^+_i - x_i|$ increases with respect to i the next equation must be satisfied:

$$\frac{|x^+_i - x_i|}{|x^+_{i+1} - x_{i+1}|} = \frac{\left|\prod_{k=i}^{n}\frac{L_{k+1,k}}{L_{kk}}\right||x^+_{n+1}|}{\left|\prod_{k=i+1}^{n}\frac{L_{k+1,k}}{L_{kk}}\right||x^+_{n+1}|} = \left|\frac{L_{i+1,i}}{L_{ii}}\right| < 1.$$

The last expression is true, due to (4) and (2).

■

Corollary 1. *The function* $\gamma_{\nu j} = \left|\prod_{k=n-j+1}^{n}\frac{L_{k+1,k}}{L_{kk}}\right|$ *converges linearly to* 0 *with respect to* j.

Proof: In order to prove that $\gamma_{\nu j}$ converges linearly to 0, it is needed to demonstrate that $\lim\limits_{j\to\infty}\frac{|\gamma_{\nu,j+1}|}{|\gamma_{\nu,j}|} \in (0,1)$ [2]. By using the definition of L in (4):

$$\begin{aligned}
\lim_{j\to\infty}\frac{|\gamma_{\nu,j+1}|}{|\gamma_{\nu,j}|} &= \lim_{j\to\infty}\frac{\left|\prod\limits_{k=n-j}^{n}\frac{\alpha}{\lambda_1}\sqrt{\frac{1-(\lambda_2/\lambda_1)^{n-k}}{1-(\lambda_2/\lambda_1)^{n-k+2}}}\right|}{\left|\prod\limits_{l=n-j+1}^{n}\frac{\alpha}{\lambda_1}\sqrt{\frac{1-(\lambda_2/\lambda_1)^{n-l}}{1-(\lambda_2/\lambda_1)^{n-l+2}}}\right|}, \\
&= \lim_{j\to\infty}\frac{\left|\left(\frac{\alpha}{\lambda_1}\right)^{j+1}\prod\limits_{k=n-j}^{n}\sqrt{\frac{1-(\lambda_2/\lambda_1)^{n-k}}{1-(\lambda_2/\lambda_1)^{n-k+2}}}\right|}{\left|\left(\frac{\alpha}{\lambda_1}\right)^{j}\prod\limits_{l=n-j+1}^{n}\sqrt{\frac{1-(\lambda_2/\lambda_1)^{n-l}}{1-(\lambda_2/\lambda_1)^{n-l+2}}}\right|}, \\
&= \left|\frac{\alpha}{\lambda_1}\right|\lim_{j\to\infty}\left|\sqrt{\frac{1-(\lambda_2/\lambda_1)^{j}}{1-(\lambda_2/\lambda_1)^{j+2}}}\right|;
\end{aligned}$$

the last expression is the product of two values between 0 and 1 $\therefore$ the result is less than 1 and the rate of convergence is linear. ■

The theorem 2 gives us a chance to approximate x^+ with the first $n-j$ terms of x and the last j terms of x^+, i.e. $\tilde{x}^+ \approx [x_1, \ldots, x_{n-j}, x^+_{n-j+1}, \ldots, x^+_n]$. In other words, given the solution of $Ax = b$, an approximation of the expanded system $A^+x^+ = b^+$ is given by the computation of the last j terms of x^+.

Theorem 2 and corollary 1 show that the error between x^+_{n-j} and x_{n-j} decreases linearly with respect to j, as a consequence, the approximation of the expanded system needs few elements and the computation can be done in constant time.

III. Experimental results

In this section the proposed algorithm is tested with three experiments, the first one is a simple example that shows an easy computation of our proposed method, the second one is a real time interpolation of an ECG and a comparison with cubic B-Spline interpolation, and the third one is related to theoretical properties of our method.

A. First numerical example

The following numerical example shows the first validation of the proposed method, by defining two linear systems:

$$\begin{bmatrix}4&1&0&0\\1&4&1&0\\0&1&4&1\\0&0&1&4\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix} = \begin{bmatrix}3\\1\\1\\2\end{bmatrix} \quad \begin{bmatrix}4&1&0&0&0\\1&4&1&0&0\\0&1&4&1&0\\0&0&1&4&1\\0&0&0&1&4\end{bmatrix}\begin{bmatrix}y_1\\y_2\\y_3\\y_4\\y_5\end{bmatrix} = \begin{bmatrix}3\\1\\1\\2\\4\end{bmatrix};$$

with solutions

$$\begin{aligned}
x &= [0.7416, 0.0335, 0.1244, 0.4689], \\
y &= [0.7462, 0.0154, 0.1923, 0.2154, 0.9462], \\
|y - x| &= [0.0045, 0.0181, 0.0679, 0.2535];
\end{aligned}$$

where the two systems share the matrix structure and the same right side results (except for the last value in the second system). Because the second system grows from the first one, it is expected that both solutions x and y are close each other. According to theorem 2, can be deduced that the absolute values $|y_5 - x_5|, \ldots, |y_1 - x_1|$ are in decreasing order and tends to zero as expected.

B. ECG interpolation

Real time interpolation is a common task in data acquisition systems as in an ECG. The cubic B-Spline interpolation is usually applied in computer graphics because its simplicity, quick computation ($\mathcal{O}(\backslash)$) and second order continuity. Even so, in real time applications where new data arrives continuously, there is no sufficiently time to perform the interpolation between samples.

Cubic B-Spline interpolation uses the matrix defined in (4) with $\alpha = 1$ and $\beta = 4$, therefore, it is possible to use the proposed method to perform the approximation of a cubic B-Spline interpolation in real time.

In this example ECG data downloaded from physionet[1] is used:

- Database: MIT-BIH Long-Term ECG Database
- Record: 14046
- Sex: Male

[1] http://physionet.org/cgi-bin/atm/ATM?database=ltdb&record=14046&tdur=3600

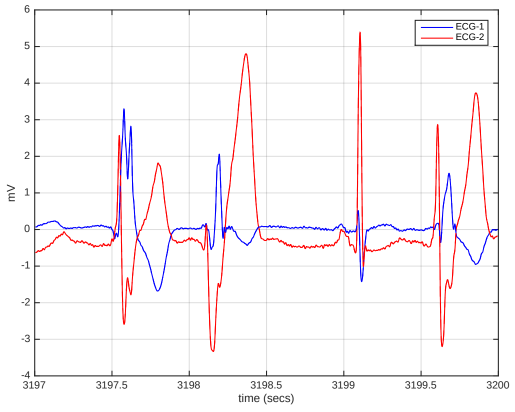

Fig. 1. Cubic B-Spline interpolation of an ECG performed in real time. Average error is about 10^{-6} in both cases.

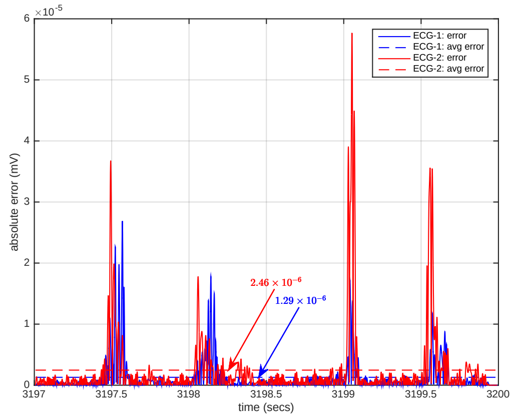

Fig. 2. Cubic B-Spline interpolation of an ECG performed in real time. Average error is about 10^{-6} in both cases.

- Age: 46
- Signals: 2
- Length: 1 hr
- Sampling frequency: 128 Hz
- Sampling interval: 0.0078125 sec
- Samples: 460 800

Because data were collected in 1 hr, it is not a good idea to wait 1 hr to perform cubic interpolation, moreover, when solving the full system every time a data arrives a $\mathcal{O}(n)$ method becomes $\mathcal{O}(n^2)$, i.e. as the system size increases, the computation time increases quadratically. Because our method is $\mathcal{O}(1)$, any approximation is computed in a constant time, and furthermore, for every new data arriving the algorithm becomes $\mathcal{O}(n)$. Computer experiments indicate that solving a system with 460800 elements in size is computed in 0.061 secs using the Cholesky method, this time is greater than the sampling interval, while the proposed method takes 10^{-6} secs independently of the system size and can be used for real time applications.

Interpolation of an ECG is shown in Fig. 1 within an interval from 3197s to 3200s with 383 samples. Cubic B-Spline interpolation was computed and an approximation by using the method described in this paper. In Fig. 1, it is not possible to observe a visual difference between the cubic B-Spline curve and its approximation using the proposed method, this fact is confirmed in Fig. 2 which shows an average error near to 10^{-6}.

Fig. 2 shows the absolute error between the cubic B-Spline interpolation and the proposed method. Cubic B-Spline approximation is near to the exact solution as expected.

Notice that in this case, the error is accumulated through n, and no necessarily is in increasing order.

Now, a comparison of the total computation time used to interpolate ECG samples with the proposed method ($\mathcal{O}(1)$) and the Cholesky method ($\mathcal{O}(n)$) is described. Fig. 3 shows 100 independent experiments of both methods and its averages. A single experiment consists of the total computation time vs system size from $n = 1$ to $n = 16000$. It is possible to observe

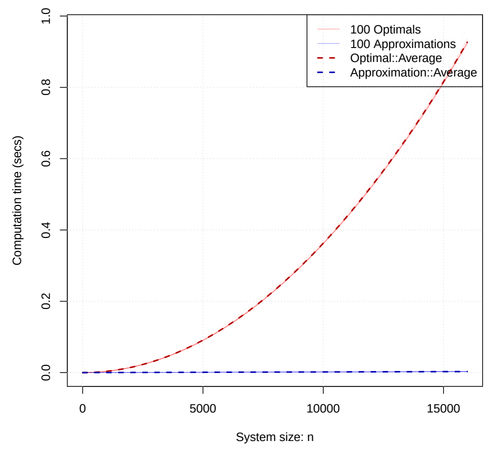

Fig. 3. This plot shows 100 independent experiments to compare the computation time between the Cholesky Method $\mathcal{O}(n)$ in blue lines and our method $\mathcal{O}(1)$ in red lines, in a scenario where the linear system grows from size 1 to 16000 and it is necessary to compute the solution every time the system grows. The cholesky method computes the optimal solution while our method computes and approximation with a relative error of 10^{-6}. In this case the Cholesky method becomes $\sum_{k=1}^{n} \mathcal{O}(k) = \mathcal{O}(n^2)$ and the proposed method becomes $\sum_{k=1}^{n} \mathcal{O}(1) = \mathcal{O}(n)$.

that Cholesky method grows quadratically and the proposed method grows linearly, because $\sum_{k=1}^{n} \mathcal{O}(k) = \mathcal{O}(n^2)$ and $\sum_{k=1}^{n} \mathcal{O}(1) = \mathcal{O}(n)$.

C. Plots and surfaces of the theoretical properties

The matrix A in (4), can be described with a real value because A has one degree of freedom; equation (9) implies $|T(A)| < 1$, $\forall A$ with the structure shown in equation (4).

$$T : \mathbb{R}^{n\times n} \mapsto \mathbb{R} \qquad T(A; \alpha, \beta) = 2\frac{\alpha}{\beta} \tag{9}$$

Fig. 4 shows the log surface of the function proportional to the absolute error between x^+ and x, i.e. $|x^+ - x| \propto \gamma_{\nu j} = \prod_{k=n-j+1}^{n} \left|\frac{L_{k+1,k}}{L_{k,k}}\right|$, in this plot, the j axis represents the number

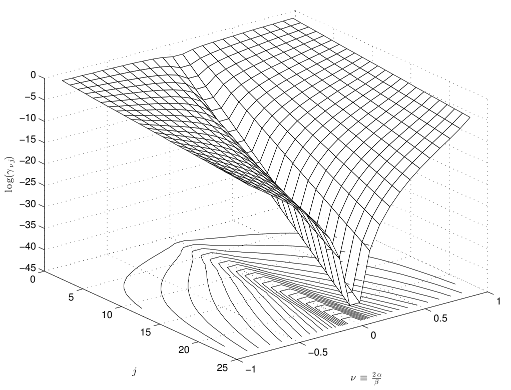

Fig. 4. Log surface of the function $\gamma_{\nu j}$ related to the error of the current and previous solution x^+ and x, respectively; see theorem 2. The parameter j is the index in reverse order that compares elements of x^+ and x, while the parameter $\nu = 2\alpha/\beta$ is related to the matrix structure through the transformation (9). Notice that the error decreases when ν is near to zero, i.e. where the matrix A becomes more diagonally dominant.

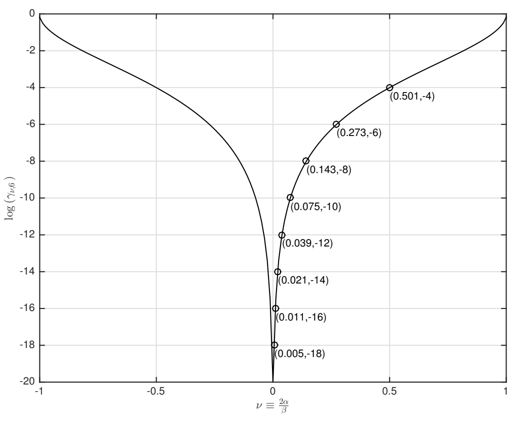

Fig. 5. Slice of the function $\gamma_{\nu j}$ at $j = 6$, see Fig. 4. Marks are located at values where $x^+ - x$ is proportional to 10^{-k}, for $k = 4, 6, \ldots, 18$.

of elements counted from the end, ν is the real value that represents any matrix A through the transformation (9).

Properties related to Fig. 4 are shown bellow:

- $|x_s^+ - x_s| > |x_t^+ - x_t|$ if and only if $s > t$, as seen along the j axis;
- for a fixed ν, the rate of convergence is almost linear;
- for a fixed j, the speed of convergence of $\gamma_{\nu j}$ grows (i.e. $\log \gamma_{\nu j}$ is more negative) when ν goes to zero, indicating that the matrix A is almost diagonal, in this case the computation becomes straightforward.

Fig. 5 shows a slice of the surface in Fig. 4 for $j = 6$, i.e., considering the comparison between the last 6 elements of x^+ and x. Cubic B-Spline and Cubic Bezier interpolation uses a matrix A with $\alpha = 1$ and $\beta = 4$, which gives $\nu = 1/2$ and at this value, the error is about 10^{-4}, see Fig. 5. Table I shows the relative error when the last j terms are computed; it is necessary to compute the last 7 terms of the expanded system for a relative error less than 10^{-4}, and 11 terms are required for a relative error less than 10^{-6}.

TABLE I. NUMBER OF COMPUTATIONS AND ITS RELATIVE ERROR FOR CUBIC B-SPLINE AND CUBIC BEZIER INTERPOLATION, I.E. $\alpha = 1, \beta = 4 \rightarrow \nu = 1/2$

last j terms	Relative error
1	2.6795×10^{-1}
2	7.1797×10^{-2}
3	1.9238×10^{-2}
4	5.1548×10^{-3}
5	1.3812×10^{-3}
6	3.7010×10^{-4}
7	9.9167×10^{-5}
8	2.6572×10^{-5}
9	7.1199×10^{-6}
10	1.9078×10^{-6}
11	5.1118×10^{-7}
12	1.3697×10^{-7}
13	3.6694×10^{-8}
14	9.8069×10^{-9}
15	2.5321×10^{-9}

IV. CONCLUSION

This paper demonstrated that linear systems with special structure (4) (positive definite, tridiagonal, Toeplitz and symmetric) that grow over time, can be approximated in $\mathcal{O}(1)$ time with linear rate of convergence based on a previous solution of a smaller and similar system.

The proofs and properties of the proposed method have been shown. This approach was tested with real time interpolation of an ECG, showing that 1) the average error of the interpolation is about 10^{-6} compared with the exact solution; and 2) the computation time is constant between samples.

The proposed method $\mathcal{O}(1)$ becomes $\mathcal{O}(n)$ while traditional methods of $\mathcal{O}(n)$ becomes $\mathcal{O}(n^2)$ for real time interpolation tasks.

Due to the proposed method properties, this approach can be used in problems where data is generated in real-time environments such as handwriting recognition and interpolation of ND medical images.

ACKNOWLEDGMENT

The authors express their gratitude to the "Universidad Politécnica de Chiapas" for financing this work through the project grant: PRODEP DSA/103.5/14/10628 "Apoyo de fomento a la generación y aplicación innovadora del conocimiento".

References

[1] R. Duda, P. Hart, and D. Stork, *Pattern classification*, ser. Pattern Classification and Scene Analysis: Pattern Classification. Wiley, 2001.

[2] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, Aug. 2000.

[3] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes in C (2nd ed.): the art of scientific computing*. New York, NY, USA: Cambridge University Press, 1992.

[4] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, ser. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1996.

[5] F. Diele and L. Lopez, "The use of the factorization of five-diagonal matrices by tridiagonal Toeplitz matrices," *Applied Mathematics Letters*, vol. 11, no. 3, pp. 61–69, 1998.

[6] D. Fischer, G. Golub, O. Hald, C. Leiva, and O. Widlund, "On fourier-toeplitz methods for separable elliptic problems," *Mathematics of Computation*, vol. 28, no. 126, pp. 349–368, 1974.

[7] G. D. Smith, *Numerical Solution of Partial Differential Equations*, 2nd ed. Clarendon Press: Oxford, 1978.

[8] J. Stewart, *Calculus: Early Transcendentals*, ser. Textbooks Available with Cengage YouBook Series. Cengage Learning, 2010.

[9] O. Axelsson, *Iterative Solution Methods*. Cambridge University Press, 1996.

[10] A. B. Boudewijn, E. Bell, and B. R. Haverkort, "Serial and parallel out-of-core solution of linear systems arising from generalised stochastic petri nets," pp. 22–26, 2001.

[11] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, ser. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1996.

[12] P. Dewilde and K. Diepold, "Large-Scale Linear Computations with Dedicated Real-Time Architectures," in *Advances in Real-Time Systems*, S. Chakraborty and J. Eberspächer, Eds. Springer Berlin Heidelberg, pp. 41–81, 2012.

[13] C. Gerald, *Applied numerical analysis*. Addison-Wesley, 1980.

[14] M. James, G. Smith, and J. Wolford, *Applied numerical methods for digital computation*, 1st ed., ser. Applied Numerical Methods for Digital Computation. Harper & Row, 1985.

[15] H.-T. Yau and J.-B. Wang, "Fast bezier interpolator with real-time lookahead function for high-accuracy machining," *International Journal of Machine Tools and Manufacture*, vol. 47, no. 10, pp. 1518 – 1529, 2007.

[16] S. Chandrasekaran, M. Gu, X. Sun, J. Xia, and J. Zhu, "A Superfast Algorithm for Toeplitz Systems of Linear Equations," *SIAM Journal on Matrix Analysis and Applications*, vol. 29, no. 4, pp. 1247–1266, 2008.

[17] J. Feng, Y. Li, Y. Wang, and M. Chen, "Design of a real-time adaptive nurbs interpolator with axis acceleration limit," *The International Journal of Advanced Manufacturing Technology*, vol. 48, no. 1-4, pp. 227–241, 2010.

[18] M.-T. Lin, M.-S. Tsai, and H.-T. Yau, "Development of a dynamics-based {NURBS} interpolator with real-time look-ahead algorithm," *International Journal of Machine Tools and Manufacture*, vol. 47, no. 15, pp. 2246 – 2262, 2007.

[19] S. Noschese, L. Pasquini, and L. Reichel, "Tridiagonal Toeplitz matrices: properties and novel applications," *Numerical Linear Algebra with Applications*, vol. 20, no. 2, pp. 302–326, 2013.

[20] L. Piegl and W. Tiller, *The NURBS Book*, ser. Monographs in Visual Communication. U.S. Government Printing Office, 1997.

[21] M.-S. Tsai, H.-W. Nien, and H.-T. Yau, "Development of a real-time look-ahead interpolation methodology with spline-fitting technique for high-speed machining," *The International Journal of Advanced Manufacturing Technology*, vol. 47, no. 5-8, pp. 621–638, 2010.

[22] Y. Wang, D. Yang, and Y. Liu, "A real-time look-ahead interpolation algorithm based on akima curve fitting," *International Journal of Machine Tools and Manufacture*, vol. 85, no. 0, pp. 122 – 130, 2014.

[23] H. Zhao, L. Zhu, and H. Ding, "A real-time look-ahead interpolation methodology with curvature-continuous b-spline transition scheme for {CNC} machining of short line segments," *International Journal of Machine Tools and Manufacture*, vol. 65, no. 0, pp. 88 – 98, 2013.

[24] W. Zheng, P. Bo, Y. Liu, and W. Wang, "Fast B-spline curve fitting by L-BFGS," *Comput. Aided Geom. Des.*, vol. 29, no. 7, pp. 448–462, Oct. 2012.

[25] G. Strang, *Linear Algebra and Its Applications*. Brooks Cole, Feb. 1988.