An International Journal
computers & mathematics with applications

PERGAMON Computers and Mathematics with Applications 46 (2003) 493–499
www.elsevier.com/locate/camwa

Matrix Structure and Loss-Resilient Encoding/Decoding

V. Y. PAN
Mathematics and Computer Science Department
Lehman College, CUNY
Bronx, NY 10468, U.S.A.
vpan@lehman.cuny.edu

(Received March 2001; accepted April 2001)

Abstract—The known deterministic algorithms for loss-resilient encoding/decoding involve computations with Cauchy matrices but only weakly exploit the matrix structure. We propose several modifications with more extensive use of the matrix structure to accelerate the computations substantially.

Keywords—Loss-resilient encoding/decoding, Cauchy matrices, Vandermonde matrices, Structured matrix computations.

1. INTRODUCTION

Loss-resilient encoding/decoding is an important practical subject. The loss-resilient codes are dramatically simpler than the error-correcting codes and involve much simpler computations. Substantial additional simplification can be achieved by using randomization, but randomization is prohibitive in some applications. In this case, some practical algorithms rely on computations with Cauchy matrices. Exploitation of the matrix structure may enable dramatic speed up but the current methods use this chance only partly. In particular, these methods do not exploit several known techniques for computations with structured matrices. We elaborate on some specific improvements based on better techniques for using the Cauchy matrix structure and on shifting to the Vandermonde matrix structure.

We organize our presentation as follows. In the next section, we will recall some definitions and basic auxiliary results on computations with structured matrices and some associated polynomials. In Section 3, we will outline the generic deterministic algorithm for loss-resilient encoding/decoding. In Section 4, we will describe the known Cauchy matrix implementations of this algorithm, and in Section 5, will show their computational cost estimates. In Sections 6

Some results of this paper have been presented at the ACM Annual International Symposium on Symbolic and Algebraic Computation (ISSAC'2000), St. Andrew's, Scotland, in August 2000 and in [1, Sect. 3.7].
Supported by NSF Grant CCR 9732206 and PSC-CUNY Award 62435-0031.
The problem of using matrix structure for the improvement of the loss-resilient encoding/decoding and reference [2, Sect. 3.7] were brought to my attention by Marek Karpinski, together with his valuable comments on some implementation aspects of the known algorithms.

PII: S0898-1221(03)00226-8

and 7, we will present two of our new modifications of this approach based on better exploitation of Cauchy matrix structure, and we will also estimate the computational cost of application of these modifications. In Section 8, we will similarly cover our alternative approach where we use a Vandermonde matrix as a generator matrix. In Section 9, we will summarize our results and give some further remarks.

2. DEFINITIONS AND AUXILIARY RESULTS ON MATRIX AND POLYNOMIAL COMPUTATIONS

Hereafter, we will use boldface letters for vectors and capital letters for matrices with components and entries from a fixed field $\mathbb{F}$, $\mathbf{v} = (v_i)_{i=0}^{h-1} \in \mathbb{F}^h$, $M = (m_{i,j})_{i=0,j=0}^{h-1,l-1} \in \mathbb{F}^{h\times l}$. $\mathbf{e}^{(i)}$ is the i^{th} coordinate vector. $I_h = (\mathbf{e}^{(i)})_{i=0}^{h-1}$, $0_{h,l}$, and $R_h = (\mathbf{e}^{(i)})_{i=h-1}^{0}$ will denote the $h\times h$ identity matrix, the $h\times l$ null matrix, and the $h\times h$ reversion matrix, respectively, $I_h\mathbf{v} = \mathbf{v}$, $R_h\mathbf{v} = (v_{h-1-i})_{i=0}^{h-1}$ for any vector $\mathbf{v} = (v_i)_{i=0}^{h-1}$. M^T and $\mathbf{v}^T$ are the transposes of a matrix M and a vector $\mathbf{v}$, respectively. $C(\mathbf{u},\mathbf{v}) = (1/(u_i - v_j))_{i=0,j=0}^{h-1,l-1} \in \mathbb{F}^{h\times l}$ is the $h\times l$ Cauchy matrix defined by two vectors $\mathbf{u} = (u_i)_{i=0}^{h-1}$ and $\mathbf{v} = (v_j)_{j=0}^{l-1}$. $V_{h,l}(\mathbf{u}) = (u_i^j)_{i=0,j=0}^{h-1,l-1} \in \mathbb{F}^{h\times l}$ is the $h\times l$ Vandermonde matrix defined by its second column $\mathbf{u}$. We will write $V(\mathbf{u})$ where $h = l$. $H(\mathbf{u}) = (u_{i,j}) \in \mathbb{F}^{h\times h}$ is the upper triangular Hankel matrix with the first column $\mathbf{u} = (u_i)_{i=0}^{h-1}$,

$$u_{i,j} = \begin{cases} u_{i+j}, & \text{for } i+j < h,\\ 0, & \text{for } i+j \ge h. \end{cases}$$

$T_\mu(\mathbf{u}) = (t_{i,j}) \in \mathbb{F}^{h\times h}$ is the μ-circulant matrix with the first column $\mathbf{u} = (u_i)_{i=0}^{h-1}$,

$$t_{i,j} = \begin{cases} u_{i-j}, & \text{for } i \ge j,\\ \mu u_{i+h-j}, & \text{for } i < j. \end{cases}$$

$(u_i)_{i=0}^{h-1} = (\mathbf{u})$ is the $h\times h$ diagonal matrix with diagonal entries $u_0,\dots,u_{h-1}$.

$$n_{\mathbf{u}}(x) = \prod_{i=0}^{h-1}(x-u_i) = x^h + \mathbf{n}_{\mathbf{u}}^T\mathbf{x}, \qquad \mathbf{x} = \left(x^i\right)_{i=0}^{h-1}, \tag{1}$$

$n_{\mathbf{u}}(x)$ is the *node polynomial*, $\mathbf{n}_{\mathbf{u}}$ is the coefficient vector of the polynomial $n_{\mathbf{u}}(x) - x^h$. Hereafter, "ops" will be our abbreviation for "field operations" in the field $\mathbb{F}$, and we will write n_h, m_h, f_h, and $v(M)$ to denote the number of ops required to compute the vector $\mathbf{n}_{\mathbf{u}}$ for a given vector $\mathbf{u} \in \mathbb{F}^h$, the coefficients of the product of two polynomials of degrees summed to at most h, the vector $(u(\omega_h^j))_{j=0}^{h-1} = (\sum_{i=0}^{h-1} u_i\omega_h^{ij})_{j=0}^{h-1}$ of the discrete Fourier transform (DFT) for a given vector $\mathbf{u} = (u_i)_{i=0}^{h-1}$ and for the vector $\mathbf{w}_h = (\omega_h^i)_{i=1}^{h-1}$ of all the h^{th} roots of 1, and the product of a matrix M by a vector, respectively. $\lceil x\rceil$ will denote the integer closest to a real x such that $x \le \lceil x\rceil$, $x = \lceil x\rceil$ for an integer x.

We have the following simple and/or well-known estimates (cf., e.g., [1]):

$$v\left((v_i)_{i=0}^{h-1}\right) = h, \tag{2}$$

$$v(H(\mathbf{u})) = m_{2h}, \qquad \text{for } \mathbf{u} = (u_i)_{i=0}^{h-1}, \tag{3}$$

$$v(V(\mathbf{w})) = f_h, \qquad \text{where } \mathbf{w} = \left(\omega_h^i\right)_{i=0}^{h-1}. \tag{4}$$

EMBEDDING LEMMA. *$v(M) \ge v(N)$ for any submatrix N of a matrix M.*

TELLEGEN'S THEOREM. *(See, e.g., [3].) $v(M) + h = v(M^T) + l$ for any matrix $M \in \mathbb{F}^{h\times l}$ with no zero rows or columns.*

THEOREM 2.1. *(See [2,4].)* $n_h \le m_h \lceil \log_2 h \rceil$.

In the following, k, p, and q will denote three integer parameters of the encoding/decoding algorithms, $0 < k \le p \le q$. We will write

$$\bar{s} = 2^{\lceil \log_2 s \rceil}, \tag{5}$$

so that

$$\log_2 \bar{s} = \lceil \log_2 s \rceil, \qquad s \le \bar{s} < 2s,$$

$\bar{s}$ is an integer power of 2, where s may stand for k, p, and q. Hereafter, ω will stand for $\omega_{\bar{q}}$, a primitive $\bar{q}^{\text{th}}$ root of 1, where $\bar{q} = 2^{\lceil \log_2 q \rceil}$, and we will use the first or both of the next assumptions.

ASSUMPTION 2.1. *Field $\mathbb{F}$ contains ω, and the elements ω^i are given for all i.*

ASSUMPTION 2.2. *Field $\mathbb{F}$ contains a nonzero element a which is not equal to ω^i for any i; furthermore, the elements $(a\omega)^i$ are given for all i, $i = 0, \dots, \bar{q} - 1$.*

We have the following estimates (cf., e.g., [1]).

THEOREM 2.2. *$f_q \le 1.5q \log_2 q$ under Assumption 2.1 provided that $q = \bar{q}$ is an integer power of 2.*

THEOREM 2.3. *$m_q \le 3f_{\bar{q}} + 2\bar{q} < 9q \log_2(2q) + 4q$ under Assumption 2.1.*

THEOREM 2.4. *$v(T_{\nu,k}(\mathbf{u})) \le 3f_{\bar{k}} + 2\bar{k} + 3\bar{k}\delta(\nu)$ provided that $\nu \ne 0$, $\delta(\nu) = 0$ for $\nu = 1$, $\delta(\nu) = 1$ otherwise; $\nu \in \mathbb{F}$, k is a positive integer, $\mathbf{u} \in \mathbb{F}^k$, $q = k$, and Assumption 2.1 holds.*

We also have the following simple estimate.

THEOREM 2.5. *$v(C(\mathbf{c}, \mathbf{d})) \le 1.5\bar{p} \log_2 \bar{p} + 2f_{\bar{q}} + q$ provided that Assumptions 2.1 and 2.2 as well as the following equations hold:*

$$\mathbf{c} = \left(\omega^i\right)_{i=0}^{q-1}, \qquad \mathbf{d} = \left(a\omega^j\right)_{j=0}^{p-1}. \tag{6}$$

PROOF. The theorem is supported by the following algorithm for the computation of the vector $C(\mathbf{c}, \mathbf{d})\mathbf{v} = (\sum_{j=0}^{p-1} v_j/(c_i - d_j))_{i=0}^{q-1}$.

(1) Recursively sum pairwise the partial fractions $v_j/(x - d_j)$ in such an order that in every h^{th} recursive step the denominator in the sum equals a monic binomial of the form $x^l - a^l\omega^{gl}$, $l = 2^h$ for an integer g. (To achieve this where $p < \bar{p}$, we allow simultaneous multiplication of both numerators and denominators of some partial fractions involved by appropriate monic binomials of the above form.) Stop when there remains only a single partial fraction $u(x)/(x^{\bar{q}} - a^{\bar{q}}\omega^{\bar{q}})$.

(2) Compute and output the value of this partial fraction at $x = c_i$. The claimed bound on the overall computational cost is now verified by inspection. ∎

3. GENERIC ALGORITHM FOR LOSS-RESILIENT ENCODING/DECODING

INPUT: a field $\mathbb{F}$, two positive integers p and q, $q \ge p$, a p-packet message vector $\mathbf{m} \in \mathbb{F}^p$, and a $q \times p$ generator matrix $G \in \mathbb{F}^{q \times p}$ whose all $p \times p$ submatrices are nonsingular.

ENCODING: compute the vector $\mathbf{s} = G\mathbf{m} \in \mathbb{F}^q$ of sent messages.

TRANSMISSION (with partial loss of messages): partition the vector $\mathbf{s}$ into two subvectors $\mathbf{r} \in \mathbb{F}^p$ of received messages and $\mathbf{l} \in \mathbb{F}^{q-p}$ of lost messages and partition, respectively, the matrix G into two submatrices $R \in \mathbb{F}^{p \times p}$ and $L \in \mathbb{F}^{(q-p) \times p}$.

DECODING: recover and output the vector $\mathbf{m} = R^{-1}\mathbf{r}$ from the nonsingular linear system $R\mathbf{m} = \mathbf{r}$ of p equations.

4. CAUCHY MATRIX IMPLEMENTATION

The customary choice for the generator matrix G is $\begin{pmatrix} I_p \\ C(\mathbf{c},\mathbf{d}) \end{pmatrix}$ for two fixed vectors $\mathbf{c} \in \mathbb{F}^{q-p}$ and $\mathbf{d} \in \mathbb{F}^p$, having a total of q distinct components. Under such a choice, the two subvectors of the sent vector $\mathbf{s} \in \mathbb{F}^q$, made of its first p and its last $q-p$ components, are projected into the two subvectors $\mathbf{h} \in \mathbb{F}^k$ (head) and $\mathbf{t} \in \mathbb{F}^{p-k}$ (tail) of the received vector $\mathbf{r} \in \mathbb{F}^p$, respectively, and the matrix R is partitioned similarly, that is, $R = \begin{pmatrix} H \\ T \end{pmatrix}$, where $H = (I_{p-k}\ 0_{p-k,k})\,P \in \mathbb{F}^{(p-k)\times p}$, $P \in \mathbb{F}^{p\times p}$ is a permutation matrix, $T = C(\mathbf{b},\mathbf{d}) \in \mathbb{F}^{k\times p}$, $\mathbf{b} \in \mathbb{F}^{p-k}$ is a subvector of the vector $\mathbf{c}$, and k can be any integer in the range from 0 to p; in some applications k is typically of the order of $p/2$.

The vector equation $H\mathbf{m} = \mathbf{h}$ identifies $\mathbf{h}$ as a subvector of the vector $\mathbf{m}$ and reduces decoding to the computation of the remaining subvector $\mathbf{u} \in \mathbb{F}^k$ of the vector $\mathbf{m}$. The partition of the vector $\mathbf{m}$ into the subvectors $\mathbf{u}$ and $\mathbf{h}$ implies the partition of the vector $\mathbf{d} \in \mathbb{F}^p$ into the two subvectors $\mathbf{f} \in \mathbb{F}^k$ and $\mathbf{g} \in \mathbb{F}^{p-k}$, and we may rewrite the vector equation $T\mathbf{m} = \mathbf{t}$ as follows:

$$C(\mathbf{b},\mathbf{f})\mathbf{u} + C(\mathbf{b},\mathbf{g})\mathbf{h} = \mathbf{t}.$$

The vector $\mathbf{u}$ is obtained from the latter vector equation, where we exploit the fact that the matrix $C(\mathbf{b},\mathbf{t})$ is nonsingular because so is every Cauchy matrix $C(\mathbf{y},\mathbf{z})$ with distinct components of the vectors $\mathbf{y}$ and $\mathbf{z}$ (cf., e.g., [5,6]). Summarizing, we have the following algorithm [2,7–10].

ENCODING: compute the vector $C(\mathbf{c},\mathbf{d})\mathbf{m} = \mathbf{s}$.

DECODING: compute the vectors

(a) $\mathbf{v} = \mathbf{t} - C(\mathbf{b},\mathbf{g})\mathbf{h}$ and
(b) $\mathbf{u} = C(\mathbf{b},\mathbf{t})^{-1}\mathbf{v}$.

OUTPUT: the vector $\mathbf{m}$ composed of its two subvectors $\mathbf{h}$ and $\mathbf{u}$.

5. COMPUTATIONAL COST OF THE KNOWN IMPLEMENTATIONS OF ENCODING AND DECODING WITH CAUCHY MATRICES

The computation requires $v(C(\mathbf{c},\mathbf{d})) + v(C(\mathbf{b},\mathbf{g})) + v(C^{-1}(\mathbf{b},\mathbf{f})) + k$ ops. The practical implementations of this algorithm use $(2q-2p-1)p$ ops for encoding, $(2p-2k-1)k + p - k$ ops at stage (a) of decoding, and about $9k^2$ ops at stage (b) of decoding. The computations rely on the straightforward multiplication of $k \times l$ matrices M by vector in $v(M) = (2k-1)l$ ops and the solution of the Cauchy nonsingular linear system of k equations in about $9k^2$ ops. The Cauchy matrix structure is ignored in the multiplications by vectors and is used only partly for the solution of the linear system.

Other known algorithms enable the encoding stage in $O(r\log^2 l)$ ops and the decoding stage in $O(p\log^2 k)$ ops, for $r = \max(p, q-p)$, $l = \min(p, q-p)$, based on the reduction of the Cauchy matrix computations to polynomial evaluation and interpolation [11,12]. The overhead constants hidden in the above "O" notation, however, are moderately large, and this makes more straightforward algorithms practically superior. We believe that more effective application of various advanced techniques available for computations with structured matrices should reverse this situation. Next, we will propose some improved encoding/decoding schemes.

6. IMPROVED ENCODING/DECODING SCHEME I

Let us use assignment (6) for the vectors $\mathbf{c}$ and $\mathbf{d}$ to endow the Cauchy matrix $C(\mathbf{c},\mathbf{d})$ with additional structure. Invoke Assumptions 2.1 and 2.2 (note that the preprocessed computation of the values ω^i and $(a\omega)^i$ for $i = 0, 1, \ldots, q-1$ does not involve the message vector $\mathbf{m}$). Then

application of the Embedding Lemma and Theorem 2.5 implies the following bound on the computational cost of encoding and also (up to k ops) of stage (a) of decoding:

$$v(C(\mathbf{b},\mathbf{g})) \le v(C(\mathbf{c},\mathbf{d})) \le 1.5\bar{p}\log_2\bar{p} + 2f_{\bar{q}} + q, \tag{7}$$

where $\log_2\bar{p} = \lceil\log_2 p\rceil$, $\log_2\bar{q} = \lceil\log_2 q\rceil$ (cf. (5)). At stage (b) of decoding, our first approach relies on the following well-known equation [6;13;14, Chapter II;15], where we use some definitions of Section 2:

$$C^{-1}(\mathbf{b},\mathbf{f}) = \left(\frac{n_{\mathbf{b}}(f_i)}{n'_{\mathbf{f}}(f_i)}\right)_{i=0}^{k-1} C(\mathbf{f},\mathbf{b}) \left(\frac{n_{\mathbf{f}}(b_i)}{n'_{\mathbf{b}}(b_i)}\right)_{i=0}^{k-1}, \tag{8}$$

for polynomial $n_{\mathbf{u}}(x)$ of equation (1). Equation (8) combined with the algorithms that support equations (2) and (4) and Theorem 2.1 and Assumption 2.2 enables performing decoding stage (b) in

$$v\left(C^{-1}(\mathbf{b},\mathbf{f})\right) \le 2n_k + 4f_{\bar{q}} + v(C(\mathbf{f},\mathbf{b})) + 6k - 2 \tag{9}$$

ops, that is, in

$$v\left(C^{-1}(\mathbf{b},\mathbf{f})\right) \le 2n_k + 6f_{\bar{q}} + 1.5\bar{p}2^{\bar{p}} + q + 6k - 2 \tag{10}$$

ops, where again $\log_2\bar{p} = \lceil\log_2 p\rceil$, $\log_2\bar{q} = \lceil\log_2 q\rceil$ (cf. (5)), n_k and f_t are bounded according to Theorems 2.1 and 2.2, and inequality (10) follows from inequalities (9) and (7). According to the computational cost estimate (10), the new algorithm is clearly superior to the practical algorithms even for moderately large integers k, p, and q unless k is much smaller than p and q.

7. IMPROVED DECODING SCHEMES II AND III

At stage (b) of the latter decoding scheme, two variations can be considered, based on the two following alternative inversion formulae (cf. [1,15–17]) where again we use some definitions of Section 2:

$$C^{-1}(\mathbf{b},\mathbf{f}) = \left(n'_{\mathbf{f}}(f_i)\right)_{i=0}^{k-1}\Big)^{-1} V(\mathbf{f})V^{-1}(\mathbf{b})\left(n_{\mathbf{f}}(b_i)\right)_{i=0}^{k-1}\Big), \tag{11}$$

$$V^{-1}(\mathbf{y}) = H(\mathbf{n_y})V^{\mathsf{T}}(\mathbf{y})\left(n'_{\mathbf{y}}(y_i)\right)_{i=0}^{l-1}\Big)^{-1}, \tag{12}$$

$$V^{-1}(\mathbf{y}) = R_l T_{\nu^l}\left(\mathbf{n_y} + \nu^l\mathbf{e}^{(0)}\right)V^{\mathsf{T}}(\mathbf{y})\left(n'_{\mathbf{y}}(y_i)\left(\nu^l - y_i^l\right)_{i=0}^{l-1}\right)^{-1}, \tag{13}$$

where $\mathbf{y} = (y_i)_{i=0}^{l-1} \in \mathbb{F}^l$, $\nu \in \mathbb{F}$.

We apply the algorithms supporting the estimates of Section 2 (cf., equations (1)–(6), Embedding Lemma, Tellegen's Theorem, and Theorems 2.1–2.5) to perform decoding stage (b) by using

$$v\left(C^{-1}(\mathbf{b},\mathbf{f})\right) \le 2n_k + 5f_{\bar{q}} + m_{2k} + 5k - 2 \tag{14}$$

ops based on equations (11) and (12) for $\mathbf{y} = \mathbf{b}$ and $l = k$, and by using

$$v\left(C^{-1}(\mathbf{b},\mathbf{f})\right) \le 2n_k + 5f_{\bar{q}} + 3f_k + 2\bar{k} + 7k - 1 \tag{15}$$

ops based on equations (11) and (13) for $\mathbf{y} = \mathbf{b}$ and $l = k$. (Here again, $\log_2\bar{q} = \lceil\log_2 q\rceil$, $\log_2\bar{k} = \lceil\log_2 k\rceil$ (cf. (5)).) The bounds (10), (14), and (15) are close to each other. Their comparison depends on comparison of the magnitudes of k, p, and q.

8. THE VANDERMONDE MATRIX APPROACH

In our yet alternative approach, we rely on a Vandermonde matrix $V_{q,p}(\mathbf{c})$ (for $\mathbf{c} = (\omega^i)_{i=0}^{q-1}$) as the generator matrix H. In this case, we only require Assumption 2.1 (but not Assumption 2.2), and encoding amounts to the computation of the vector $\mathbf{s} = V_{q,p}(\mathbf{c})\mathbf{m}$, which requires

$$v(V_{q,p}(\mathbf{c})) \le f_{\bar{q}} \tag{16}$$

ops (cf. (4) and the Embedding Lemma). Decoding amounts to the solution of the linear system

$$R\mathbf{m} = \mathbf{r},$$

where R is a $p \times p$ submatrix $V(\mathbf{a})$ of the matrix $V_{q,p}(\mathbf{c})$ for a subvector $\mathbf{a} \in \mathbb{F}^p$ of the vector $\mathbf{c}$. To compute the vector $\mathbf{m} = V^{-1}(\mathbf{a})\mathbf{r}$, we apply equations (12) or (13) and the already cited algorithms that support the estimates of Section 2. Thus, we perform decoding by using

$$v\left(V^{-1}(\mathbf{a})\right) \le m_{2p} + n_p + f_{\bar{q}} + 2p - 1 \tag{17}$$

ops based on equation (12) for $\mathbf{y} = \mathbf{a}$ and $l = p$ and by using

$$v\left(V^{-1}(\mathbf{a})\right) \le n_p + f_{\bar{q}} + 3f_{\bar{p}} + 3p \tag{18}$$

ops based on equation (13) where $\mathbf{y} = \mathbf{a}$, $l = p$, $\nu = 1$, $\log_2 \bar{p} = \lceil \log_2 p \rceil$, $\log_2 \bar{q} = \lceil \log_2 q \rceil$. (We assume that the values $1 - y_i^l$ have been precomputed for all i.) Comparing bounds (16)–(18) with (10), (14), and (15), we observe a substantial advantage of the approach of this section, at least unless k is much less than p.

9. CONCLUSION

In the known deterministic algorithms for loss-resilient encoding/decoding, we improved substantially the stage of computations with structured matrices. Other techniques used in the known algorithms (such as replacing ops in a field $\mathbb{F}$ with operations with polynomials over a smaller field or with XORs of computer words [2]) can be incorporated in our algorithms too. Our algorithms substantially speed up the known algorithms at least unless the parameters k and p are much less than q. Our approach relies on the assumption that the computations can be performed in a field with $\bar{q}^{\text{th}}$ roots of 1 where $\bar{q}$ is a power of 2, $\bar{q} = 2^{\lceil \log_2 q \rceil}$, $q \le \bar{q} < 2q$. Simple modification can be applied in the fields with $2^{h\,\text{th}}$ roots of 1 where $n/2^h$ is not large by partitioning the generator matrix G into $2^h \times 2^h$ blocks. More generally, our approach can be extended to any field containing $q + 1$ distinct powers $1, a, a^2, \ldots, a^q$ of some element a. In this case, the basic operation of discrete Fourier transform (DFT) should be replaced by a little slower operation of generalized DFT (cf. [1, p. 29; 17, p. 14]), which would slow down the computation a little (by a constant factor) but may still keep it effective, depending on the magnitudes of the parameters k, p, and q. The reader is referred to [18,19] on specific estimates for the computational cost of this approach, as well as the hybrid algorithms, which combine our approach with incorporation of some blocks of straightforward matrix-by-vector products.

REFERENCES

1. V.Y. Pan, *Polynomial and Structured Matrix Computations: Unified Superfast Algorithms*, Birkhäuser/ Springer, Boston, MA/New York, NY, (2001).
2. J. Blömer, M. Kalfane, R. Karp, M. Karpinski, M. Luby and D. Zuckerman, An XOR-based erasure-resilient coding scheme, Technical Report TR-95-48, International Computer Science Institute, Berkeley, CA, (1995).
3. P. Penfield Jr., R. Spencer and S. Duinker, *Tellegen's Theorem and Electrical Networks*, MIT Press, Cambridge, MA, (1970).
4. J. von zur Gathen and J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, Cambridge, U.K., (1999).
5. L. Mirsky, *An Introduction to Linear Algebra*, Dover, New York, (1982).
6. M.O. Rabin, Efficient dispersal of information for security, load balancing, and fault tolerance, *J. ACM* **36** (2), 335–348, (1989).
7. F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, New York, (1977).
8. A. Albanese, J. Blömer, J. Edmonds, M. Luby and M. Sudan, Priority encoding transmission, In *Proc. 35th Ann. Symp. on Foundations of Computer Science (FOCS)*, pp. 604–613, IEEE Computer Society Press, (1994).
9. S.B. Wicker and V. Bhargava, *Reed-Solomon Codes and Their Applications*, IEEE Press, New York, (1994).

10. N. Alon, J. Edmonds and M. Luby, Linear time erasure codes with nearly optimal recovery, In *Proc. 36th Ann. Symp. on Foundations of Computer Science (FOCS)*, pp. 512–519, IEEE Computer Society Press, (1995).
11. N. Gastinel, Inversion d'une matrice generalisant la matrice de Hilbert, *Chiffres* **3**, 149–152, (1960).
12. A. Gerasoulis, M.D. Grigoriadis and L. Sun, A fast algorithm for Trummer's problem, *SIAM J. Sci. Statist. Comput.* **8** (1), 135–138, (1987).
13. D.E. Knuth, *The Art of Computer Programming, Volume 1*, Addison-Wesley, Reading, MA, (1968).
14. H. Heinig and K. Rost, Algebraic methods for Toeplitz-like matrices and operators, *Operator Theory* **13**, Birkhäuser, (1984).
15. T. Fink, G. Heinig and K. Rost, An inversion formula and fast algorithms for Cauchy-Vandermonde matrices, *Linear Algebra Appl.* **183**, 179–191, (1993).
16. I. Gohberg and V. Olshevsky, Fast algorithms with preprocessing for matrix-vector multiplication problem, *J. of Complexity* **10** (4), 411–427, (1994).
17. D. Bini and V.Y. Pan, *Polynomial and Matrix Computations, Volume 1: Fundamental Algorithms*, Birkhäuser, Boston, MA, (1994).
18. V.Y. Pan, Matrix structure, polynomial arithmetic and erasure resilient encoding/decoding, In *Proc. Intern. Symp. on Symbolic and Algebraic Computations (ISSAC'2000)*, pp. 266–271, ACM Press, New York, (2000).
19. V.Y. Pan, Improving loss-resilient encoding/decoding by exploiting matrix structure, Preprint, (2000).
20. M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, D.A. Spielman and V. Stemann, Practical loss-resilient codes, In *Proc. 29th Ann. Symp. on Theory of Computing (STOC'97)*, ACM Press, New York, (1997).