Pergamon

Mathl. Comput. Modelling Vol. 19, No. 12, pp. 85–93, 1994

0895-7177/94 $7.00 + 0.00

0895-7177(94)00083-2

An Error-Free Algorithm to Solve a Linear System of Polynomial Equations

M. MORHÁČ
Institute of Physics, Slovak Academy of Sciences
Bratislava, Slovakia

(Received December 1992; accepted October 1993)

Abstract—The paper presents an error-free algorithm to solve a system of linear equations with polynomial coefficients. Modular arithmetic in residual polynomial class and in residual numeric class is employed. The algorithm is iterative and well suited for implementation for computers with vector operations and fast and error-free convolutors.

Keywords—Linear polynomial system, Residual polynomials, Inverse polynomial, Roundoff errors, Modular arithmetic.

1. INTRODUCTION

In recent years, increased attention has been paid to problems of error-free numerical algorithms. These cover questions of error-free convolution [1], deconvolution [2–4] Volterra system identification [5] or error-free solution of general systems of linear equations employing parallel algorithm [6,7], or sequential algorithm derived in [8,9].

The task of this paper is to find the algorithm of error-free solution of linear system

$$A(s) \cdot \bar{x}(s) = \bar{y}(s), \tag{1}$$

where the elements of polynomial matrix A and output vector are

$$a_{i,j}(s) = {}^{p-1}a_{i,j} \cdot s^{p-1} + \cdots + {}^{1}a_{i,j} \cdot s^{1} + {}^{0}a_{i,j}, \tag{2}$$

$$y_i(s) = {}^{p-1}y_i \cdot s^{p-1} + \cdots + {}^{1}y_i \cdot s^{1} + {}^{0}y_i, \tag{3}$$

and $i, j \in \langle 0, n-1 \rangle$. The dimension of matrix $A(s)$ and the length of vector $\bar{y}(s)$ is determined by n and maximum degree of polynomials $a_{i,j}(s)$ and $y_i(s)$ is determined by parameter p. Application of some of classic algorithms, e.g., Gauss-Jordan's elimination, leads to very large degrees (n.p) of polynomials of inverse matrix $A^{-1}(s)$ and determinant of matrix $A(s)$. If coefficients of polynomials $a_{i,j}(s)$ are represented in the third dimension of matrix, then memory requirements for elements of matrix $A^{-1}(s)$ become large. As well if the system is ill-conditioned the roundoff errors of coefficients of inverse matrix $A^{-1}(s)$ can negatively influence solution.

Presented method restricts the degree of polynomial elements during calculation to constant value (p) and, therefore, restricts memory space needed for calculation $A^{-1}(s)$. All operations with polynomials are carried out in residual polynomial class modulo $(s^p - 1)$. The reason why we have chosen this class of residual polynomials is explained in Part 4. Also all numerical operations with coefficients are carried out in modulo M arithmetic what removes roundoff errors from the solution.

2. MATHEMATICAL BACKGROUND

Assuming the coefficients being integers, the solution of the system given by (1), (2), (3) can be expressed as

$$\bar{x}(s) = \frac{1}{D(s)} \cdot [(s^p - 1)^0 \cdot \bar{x}_0(s) + (s^p - 1)^1 \cdot \bar{x}_1(s) + \cdots + (s^p - 1)^k \cdot \bar{x}_k(s)], \tag{3a}$$

and vectors $\bar{x}_i$, $i = 0, 1, \ldots, k$ are:

$$\bar{x}_i(s) = [M^0 \cdot \bar{x}_{i0}(s) + M^1 \cdot \bar{x}_{i1}(s) + \cdots + M^m \cdot \bar{x}_{im}(s)], \tag{4}$$

where M is prime modulus, D is the determinant of matrix A and m, k are finite integers. Let us suppose for the moment that the determinant $D(s)$ (polynomial) is known. We denote polynomial modulus as

$$P(s) = s^p - 1. \tag{5}$$

Then substituting (3) into (1) gives

$$\bar{y} = \frac{1}{D(s)} \cdot [P^0(s) \cdot A(s) \cdot \bar{x}_0(s) + P^1(s) \cdot A(s) \cdot \bar{x}_1(s) + \cdots + P^k(s) \cdot A(s) \cdot \bar{x}_k(s)], \tag{6}$$

and

$$\frac{1}{P(s)} \cdot \left[\cdots \frac{1}{P(s)} \cdot \left[\frac{1}{P(s)} (\bar{y}(s) \cdot D(s) - A(s) \cdot \bar{x}_0(s)) - A(s) \cdot \bar{x}_1(s) \right] - \cdots A(s) \cdot \bar{x}_m(s) \right] = 0. \tag{7}$$

Comparing (6), (7) to analogous relations in [2,9], it is obvious that formally the algorithm presented in these papers can be also used to solve linear polynomial system. One of the differences is that instead of numerical modulus the polynomial modulus must be used. Therefore, the algorithm is introduced without proof.

Algorithm I

a. Let $\overline{yp}_0(s) = \bar{y}(s)$.
b. Calculate vector $\bar{x}_p(s)$

$$\bar{x}_p(s) = A^{-1}(s) \cdot \overline{yp}_0(s) \ (\text{mod } P(s)). \tag{8}$$

c. Let

$$\bar{x}_0(s) = D_p(s) \cdot \bar{x}_p(s) \ (\text{mod } P(s)), \tag{9}$$

where determinant of $A(s)$ is:

$$D(s) = t \cdot P(s) + D_p(s), \tag{10}$$

i.e.,

$$D_p(s) = D(s) \ (\text{mod } P(s)).$$

d. Let $\bar{y}_0(s) = \bar{y}(s) \cdot D(s)$.
e. Let $\bar{x}(s) = 0$.
f. Let $j = 0$.
g. Let $\bar{x}(s) = \bar{x}(s) + \bar{x}_j(s) \cdot P^j(s)$.
h. Calculate:

$$\bar{y}'_j(s) = A(s) \cdot \bar{x}_j(s), \tag{11}$$

$$\bar{y}_{j+1}(s) = \frac{1}{P(s)} \cdot (\bar{y}_j(s) - \bar{y}'_j(s)), \tag{12}$$

$$\overline{yp}_{j+1}(s) = \bar{y}_{j+1}(s) \ (\text{mod } P(s)). \tag{13}$$

i. If $\bar{y}_{j+1}(s)$ equals zero vector, finish the calculation. The solution is vector $\bar{x}(s)$ and determinant $D(s)$.

j. $j = j + 1$.

k.

$$\bar{x}_j(s) = A^{-1}(s) \cdot \overline{yp}_j(s) \pmod{P(s)}. \tag{14}$$

g. Go to g.

The final solution is given by $\bar{x}(s)/D(s)$. All operations in Algorithm I are carried out in integers.

The degree of polynomial elements of the inverse matrix $A^{-1}(s) \pmod{P(s)}$ is restricted to p which simplifies operations (8), (9), (14) and restricts memory space for $A^{-1}(s)$. The integers of coefficients of matrix $A^{-1}(s) \pmod{P(s)}$, vectors $\bar{x}_j(s)$, $\bar{y}'_j(s)$, $\bar{y}_j(s)$, $\overline{yp}_j(s)$, $j = 0, 1, 2, \ldots$, and $D(s)$, $D_p(s)$, are very large numbers and to store each of them, one needs several computer words. This also increases memory requirements. It is possible, at the expense of precision, to store them in another format (e.g., floating point format). This can, if ill-conditioned systems, cause well-known difficulties with stability of solution and even destroy it. We have at the disposal residual arithmetic, which can be employed to calculate the polynomial coefficients, now in prime residual class M. The reasons why M is chosen to be prime are dealt with, e.g., [6–8]. This means that to calculate solutions of system (1) we need also to calculate

$$B(s) = A^{-1}(s) \cdot (\text{mod } P(s)) \cdot (\text{mod } M). \tag{15}$$

Similar considerations like in derivation of Algorithm I in residual class $P(s)$ can be, with respect to (4), used also in calculation of the vectors $\bar{x}_j(s)$, $j = 0, 1, 2 \ldots$ (steps e and g of the Algorithm I) in residual class M. Then also the requirements for precision of solution of (1) are fulfilled.

The final algorithm to solve linear system of polynomial equations in residual class $P(s)$, (polynomial coefficients are calculated in residual class M) can be defined as follows

Algorithm II

a. Let $\overline{yp}_{00}(s) = \bar{y}(s)$.

b. Calculate vector:

$$\bar{x}_p(s) = B(s) \cdot \overline{yp}_{00}(s) \cdot (\text{mod } P(s)) \cdot (\text{mod } M), \tag{16}$$

where matrix $B(s)$ is given by (15).

c. Let

$$\bar{x}_{00}(s) = D_{pm}(s) \cdot \bar{x}_p(s) \cdot (\text{mod } P(s)) \cdot (\text{mod } M), \tag{17}$$

where

$$D_{pm}(s) = D(s) \cdot (\text{mod } P(s)) \cdot (\text{mod } M).$$

d. Let $\bar{T}(s) = \bar{y}(s) \cdot D(s)$.

e. Let $\bar{x}(s) = 0$.

f. Let $j = 0$, $k = 0$.

g. Let

$$\bar{x}(s) = \bar{x}(s) + \bar{x}_{jk}(s) \cdot P^j(s) \cdot M^k. \tag{18}$$

h. Calculate:

$$\bar{y}'_{jk}(s) = A(s) \cdot \bar{x}_{jk}(s), \tag{19}$$

$$\bar{T}(s) = \bar{T}(s) - \bar{y}'_{jk}(s) \cdot M^k, \tag{20}$$

$$\bar{y}'_{j,k+1}(s) = \frac{\bar{T}(s)(\text{mod } P(s))}{M}, \tag{21}$$

$$\overline{yp}_{j,k+1}(s) = \bar{y}_{j,k+1}(s)(\text{mod } M). \tag{22}$$

i. If $\bar{y}'_{j,k+1}(s) = \bar{0}$, continue in step m.
j. $k = k + 1$.
k.

$$\bar{x}_{j,k}(s) = B(s) \cdot \overline{yp}_{j,k}(s) \cdot (\text{mod } M). \tag{23}$$

l. Go to g.
m. If $\bar{T}(s) = \bar{0}$, finish the calculation.
n. $j = j + 1$, $k = 0$.
o.

$$\bar{T}(s) = \frac{\bar{T}(s)}{P(s)}, \tag{24}$$

$\overline{yp}_{j,k}(s) = \bar{T}(s) \cdot (\text{mod } M)$.
p. Go to k.

The result is again given by $\bar{x}(s)/D(s)$.

In Algorithm II, we supposed that the inverse matrix $A^{-1}(s)$ (mod $P(s)$) (mod M) and determinant $D(s)$ are known. The algorithm of their calculation is dealt with in the following chapter.

3. CALCULATION OF AN INVERSE MATRIX AND DETERMINANT

We start again from Parts 3 or 4 of [9], where the algorithm to calculate determinant or inverse matrix in sense of residual arithmetic is derived. According to [9], we can express determinant of matrix of dimension N as

$$D_N = \prod_{j=1}^{N-1} \left[a(j,j) - \sum_{i=0}^{j-1} a(j,i) \cdot k_j(i) \right], \tag{24a}$$

where vectors $\bar{k}_j$ are the solutions of systems

$$A_{j,j} \cdot \bar{k}_j = A_{j,1}, \qquad j = 1, 2, \ldots, N-1, \tag{24b}$$

and indices denote matrix partitioning

$$A_{j+1,j+1} = \begin{bmatrix} A_{j,j} & A_{j,1} \\ A_{1,j} & a(j+1,j+1) \end{bmatrix}. \tag{24c}$$

Similarly, the inverse matrix B partitioned analogously with (24c) is obtained as

$$B_{1,1} = \left[A_{1,1} - A_{1,j} \cdot A_{j,j}^{-1} \cdot A_{j,1} \right]^{-1} (\text{mod } P(s))(\text{mod } M), \tag{24d}$$

$$B_{j,1} = -A_{j,j}^{-1} \cdot A_{j,1} \cdot B_{1,1} (\text{mod } P(s))(\text{mod } M), \tag{24e}$$

$$B_{1,j} = -B_{1,1} \cdot A_{1,j} \cdot A_{j,j}^{-1} (\text{mod } P(s))(\text{mod } M), \tag{24f}$$

$$B_{j,j} = A_{j,j}^{-1} + B_{j,1} \cdot B_{1,1}^{-1} \cdot B_{1,j} (\text{mod } P(s))(\text{mod } M), \tag{24g}$$

where $j = 1, 2, \ldots, N-1$. Coefficients of matrices are polynomials. Comparing the matrices to [9], the only difference in relations (24d)–(24g) consists in performing all operations in two residual classes, $P(s)$ and M.

The only remaining problem in (24d)–(24g) is the calculation of inverse polynomial in residual classes $P(s)$ and M.

4. CALCULATION OF AN INVERSE POLYNOMIAL

We have to find polynomial $b(s)$ to polynomial $a(s)$ so that

$$a(s) \cdot b(s) \cdot (\text{mod } P(s)) \cdot \text{mod } (M) = 1. \tag{25}$$

According to [1], the cyclic convolution using polynomial algebra concept is

$$c_\ell = \sum_{i=0}^{p-1} a_i \cdot b_{\ell-i}, \tag{26}$$

where indices $\ell - i$ are calculated modulo p. It can be expressed in the form of polynomial product as

$$a(s) \cdot b(s) \tag{27}$$

modulo $s^p - 1$. Coefficients a_i, $b_{\ell-i}$ are assigned to corresponding polynomial coefficients of $a(s)$, $b(s)$. It is known that (26) can be expressed in matrix notation

$$\begin{bmatrix} a(0), & a(p-1), \ldots, & a(1) \\ a(1), & a(0), \ldots, & a(2) \\ \vdots & \vdots & \vdots \\ a(p-1), & a(p-2) \ldots, & a(0) \end{bmatrix} \cdot \begin{bmatrix} b(0) \\ b(1) \\ \vdots \\ b(p-1) \end{bmatrix} = \begin{bmatrix} c(0) \\ c(1) \\ \vdots \\ c(p-1) \end{bmatrix}. \tag{28}$$

From what is given above, it follows that solution of equations (25) represents the solution of linear system

$$\begin{bmatrix} a(0), & a(p-1), \ldots, & a(1) \\ a(1), & a(0), \ldots, & a(2) \\ \vdots, & \vdots & \vdots \\ a(p-1), & a(p-2), \ldots, & a(0) \end{bmatrix} \cdot \begin{bmatrix} b(0) \\ b(1) \\ \vdots \\ b(p-1) \end{bmatrix} \cdot \text{mod } (M) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \tag{29}$$

in residual class M.

The equation (29) can be solved using any known algorithm. If we choose, e.g., M to be Fermat number we can use Fermat transform [2,3].

We introduce algorithm presented in [4]. According to it, we get inverse polynomial $b(s)$ by successive reductions of polynomial $a(s)$ or vector $\bar{a}$. For the $r + 1 - st$ reduction it holds

$$^{r+1}a(2^{r+1} \cdot \ell) = \sum_{i=0}^{(N/2^r)-1} {}^{r}a(2^r \cdot i) \cdot {}^{r}a(m) \cdot (-1)^i \cdot \text{mod } (M),$$

where

$$m = (2^{r+1} \cdot \ell - 2^r \cdot i) \cdot \text{mod } (N); \qquad \ell \in \left\langle 0, \left(\frac{N}{2^{r+1}} \right) - 1 \right\rangle,$$

and number of reduction $r = 0, 1, \ldots, \log_2 N$. Let

$$^{0}\bar{n} = [1, 0, \ldots, 0]^T, \quad \text{and}$$

$$^{r+1}n(\ell) = \sum_{i=0}^{(N/2^r)-1} {}^{r}n(2^r \cdot i) \cdot {}^{r}a(m) \cdot (-1)^i \cdot \text{mod } (M),$$

where

$$m = (\ell - 2^r \cdot i) \cdot \text{mod } N; \qquad \ell \in \langle 0, n-1 \rangle,$$

and number of reduction $r = 0, 1, \ldots, \log_2 N - 1$. Resulting solution of polynomial inversion is

$$\bar{b} = \frac{{}^t\bar{n}}{{}^t a(0)} \cdot \mathrm{mod}\ (M),$$

where $t = \log_2 N$. We introduced only final formulas of the algorithm. To understand details, we refer to [4].

Now we outline the algorithm using an example. We choose the parameter p to be power of 2. Let us have polynomial

$$a(s) = 2s^2 + 4s + 1 \quad \text{and} \quad p = 4,\ M = 11.$$

Then $b(s)$ can be expressed as

$$b(s) = \frac{1}{2s^2 + 4s + 1} \ (\mathrm{mod}\ 11).$$

Let us multiply numerator and denominator of this relation by polynomial $a(-s)$ (mod 11). We get

$$\begin{aligned} b(s) &= \frac{(2s^2 + 7s + 1)}{(2s^2 + 4s + 1) \cdot (2s^2 + 7s + 1)} \cdot (\mathrm{mod}\ 11) \\ &= \frac{(2s^2 + 7s + 1)}{(10s^2 + 5)} \ (\mathrm{mod}\ 11) = \frac{(2s^2 + 7s + 1)}{a_1(s^2)} \ (\mathrm{mod}\ 11). \end{aligned} \tag{30}$$

Let us call the previous operation to be reduction according to variable s. After this operation the coefficients of denominator with odd powers equal zero.

Let us continue in these reductions and multiply numerator and denominator in (30) by polynomial $a_1(-s^2)$. Then

$$b(s) = \frac{(2s^2 + 7s + 1) \cdot (s^2 + 5)}{(10s^2 + 5) \cdot (s^2 + 5)} \ (\mathrm{mod}\ 11) = \frac{7s^3 + 2s + 7}{2} \ (\mathrm{mod}\ 11).$$

Multiplying numerator by $[2^{-1}] \cdot (\mathrm{mod}\ 11) = 6$, we obtain final solution

$$b(s) = 9s^3 + s + 9.$$

Substituting this result into (25), we can check its correctness.

5. ILLUSTRATION OF AN ALGORITHM TO SOLVE LINEAR SYSTEM OF POLYNOMIALS

Let us have linear system consisting of polynomials:

$$\begin{bmatrix} s+1, & 3s \\ 2s, & s^2+1 \end{bmatrix} \cdot \bar{x}(s) = \begin{bmatrix} s^2+1 \\ 2 \end{bmatrix}. \tag{30a}$$

We use polynomial modulus

$$P(s) = s^3 - 1,$$

and numerical modulus $M = 5$.

We assume that we know (we have calculated it according to algorithms in Part III) determinant

$$D(s) = s^3 - 5s^2 + s + 1,$$

determinant in residue class $P(s)$ and residue class M

$$Dpm(s) = D(s)(\text{mod } (s^3 - 1))(\text{mod } 5) = s + 2, \tag{31}$$

and inverse matrix in residue class $P(s)$ and residue class M

$$B(s) = A^{-1}(s) \cdot (\text{mod } (s^3 - 1)) \cdot (\text{mod } 5) = \begin{bmatrix} s+3, & 4s^2+2s+3 \\ s^2+3s+2, & s^2+3s \end{bmatrix}.$$

In steps a, b in Algorithm II, we calculate $\bar{x}_p(s)$

$$\bar{x}_p(s) = \begin{bmatrix} s+3, & 4s^2+2s+3 \\ s^2+3s+2, & s^2+3s \end{bmatrix} \cdot \begin{bmatrix} s^2+1 \\ 2 \end{bmatrix} \cdot (\text{mod } (s^3-1)) \cdot (\text{mod } 5) = \begin{bmatrix} s^2 \\ 0 \end{bmatrix}.$$

Using $Dpm(s)$ from (31), we get:

$$\bar{x}_{00}(s) = Dpm(s) \cdot \begin{bmatrix} s^2 \\ 0 \end{bmatrix} (\text{mod}(s^3-1)(\text{mod } 5)) = \begin{bmatrix} 2s^2 \\ 0 \end{bmatrix}.$$

Applying steps d, e, f, g, of Algorithm II, we initialize vectors and control variables

$$\bar{T}(s) = \bar{y}(s) \cdot D(s) = \begin{bmatrix} s^2+1 \\ 2 \end{bmatrix} \cdot (s^3 - 5s^2 + s + 1) = \begin{bmatrix} s^5 - 5s^4 + 2s^3 - 4s^2 + s + 1 \\ 2s^3 - 10s^2 + 2s + 2 \end{bmatrix},$$

$j = 0$, $k = 0$,

$$\bar{x}(s) = \begin{bmatrix} 2s^2+1 \\ 0 \end{bmatrix}.$$

In step h, we calculate vectors

$$\bar{y}'_{00}(s) = A(s) \cdot \bar{x}_{00}(s) = \begin{bmatrix} s+1, & 3s \\ 2s, & s^2+1 \end{bmatrix} \cdot \begin{bmatrix} 2s^2+1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2s^3+2s^2+s+1 \\ 4s^3+2s \end{bmatrix},$$

$$\bar{T}(s) = \bar{T}(s) - \bar{y}'_{00}(s) \cdot M^0 = \begin{bmatrix} s^5 - 5s^4 - 6s^2 \\ -2s^3 - 10s^2 + 2 \end{bmatrix},$$

$$\bar{y}_{01}(s) = \frac{\bar{T}(s)\ (\text{mod}(s^3-1))}{5} = \frac{1}{5} \cdot \begin{bmatrix} -5s^2 - 5s \\ -10s^2 \end{bmatrix} = \begin{bmatrix} -s^2 - s \\ -2s^2 \end{bmatrix},$$

$$\overline{yp}_{01}(s) = \bar{y}_{01}(s)\ (\text{mod } 5) = \begin{bmatrix} 4s^2+4s \\ 3s^2 \end{bmatrix}.$$

The vector $\bar{y}_{01}(s) \neq 0$, thus we set $k = 1$ and continue in step k

$$\begin{aligned} \bar{x}_{01}(s) &= B(s) \cdot \overline{yp}_{01}(s)(\text{mod}(s^3-1))(\text{mod } 5) \\ &= \begin{bmatrix} s+3, & 4s^2+2s+3 \\ s^2+3s+2, & s^2+3s \end{bmatrix} \cdot \begin{bmatrix} 4s^2+4s \\ 3s^2 \end{bmatrix} (\text{mod}(s^3-1))(\text{mod } 5) \\ &= \begin{bmatrix} 4s \\ 0 \end{bmatrix} (\text{mod } 5) = \begin{bmatrix} -5 \\ 0 \end{bmatrix}. \end{aligned}$$

NOTE. If any coefficient of the solution is greater than $(M-1)/2$, it is considered negative and in accordance with rules of modular arithmetic its negative value is calculated by subtracting modulus M.

We continue in step g.

$$\bar{x}(s) = \begin{bmatrix} 2s^2+1 \\ 0 \end{bmatrix} + \begin{bmatrix} -s \\ 0 \end{bmatrix} \cdot 5^{-1} = \begin{bmatrix} 2s^2 - 5s + 1 \\ 0 \end{bmatrix}.$$

Further, according to step h, we calculate appropriate vectors

$$\bar{y}'_{01}(s) = a(s) \cdot \bar{x}_{01}(s) = \begin{bmatrix} s+1, & 3s \\ 2s, & s^2+1 \end{bmatrix} \cdot \begin{bmatrix} -s \\ 0 \end{bmatrix} = \begin{bmatrix} -s^2 - s \\ -2s^2 \end{bmatrix},$$

$$\bar{T}(s) = \bar{T}(s) - 5^{-1} \cdot \bar{y}'_{01}(s) = \begin{bmatrix} s^5 - 5s^4 - s^2 + 5s \\ -2s^3 + 2 \end{bmatrix},$$

$$\bar{y}_{02}(s) = \frac{\bar{T}(s) \ (\mathrm{mod}\,(s^3-1))}{5} = \bar{0},$$

$$\overline{yp}_{02}(s) = \bar{0}.$$

The vector $\bar{y}_{02}(s) = \bar{0}$, so that we continue in step m. The vector $\bar{T}(s) \neq 0$, therefore, we set control variables $j = 1$, $k = 0$. Then according to step o, we calculate

$$\bar{T}(s) = \frac{\bar{T}(s)}{s^3 - 1} = \begin{bmatrix} s^2 - 5s \\ -2 \end{bmatrix}, \quad \text{and} \quad \overline{yp}_{10}(s) = \bar{T}(s)(\mathrm{mod}\ 5) = \begin{bmatrix} s^2 \\ 3 \end{bmatrix}.$$

In step k, we calculate next particular solution

$$\bar{x}_{10}(s) = \begin{bmatrix} s+3, & 4s^2+2s+3 \\ s^2+3s+2, & s^2+3s \end{bmatrix} \cdot \begin{bmatrix} s^2 \\ 3 \end{bmatrix} \cdot (\mathrm{mod}\,(s^3-1) \cdot (\mathrm{mod}\ 5)) = \begin{bmatrix} s \\ 3 \end{bmatrix} \cdot (\mathrm{mod}\ 5) = \begin{bmatrix} s \\ -2 \end{bmatrix},$$

which we add to final solution

$$\bar{x}(s) = \begin{bmatrix} 2s^2 - 5s + 1 \\ 0 \end{bmatrix} + \begin{bmatrix} s \\ -2 \end{bmatrix} \cdot (s^3 - 1) = \begin{bmatrix} s^4 + 2s^2 - 6s + 1 \\ -2s^3 + 2 \end{bmatrix}.$$

Again according to step h, we calculate

$$\bar{y}'_{10}(s) = A(s) \cdot \bar{x}_{10}(s) = \begin{bmatrix} s+1, & 3s \\ 2s, & s^2+1 \end{bmatrix} \cdot \begin{bmatrix} s \\ -2 \end{bmatrix} = \begin{bmatrix} s^2 - 5s \\ -2 \end{bmatrix},$$

$$\bar{T}(s) = \bar{T}(s) - \bar{y}'_{10}(s) \cdot 5^0 = \bar{0},$$

$$\bar{y}_{11}(s) = \bar{0},$$

$$\overline{yp}_{11}(s) = \bar{0}.$$

The vector $\bar{y}_{11}(s)$ as well as vector $\bar{T}(s)$ equal zero vectors what means that we finish the calculation. The resulting solution is

$$\bar{x}(s) = \frac{1}{D(s)} \cdot \bar{x}(s) = \frac{1}{s^3 - 5s^2 + s + 1} \cdot \begin{bmatrix} s^4 + 2s^2 - 6s + 1 \\ -2s^3 + 2 \end{bmatrix}.$$

Substitution this solution into (30a) proves its correctness.

6. CONCLUSIONS

The presented algorithm to solve linear system of polynomials shows one possible approach to treat the problem. The use of residual polynomial class removes, in calculation of inverse matrix, the necessity to watch and store high order coefficients of polynomials. The polynomial modulus in the form $s^p - 1$ makes it possible to use any of algorithms of cyclic deconvolution which in turn simplifies calculation of inverse matrix $A^{-1}(s)$.

The calculation of coefficients of polynomials in matrix $A^{-1}(s)$ and in vector $\bar{x}(s)$ is carried out in residual class M which on one side removes the necessity to watch overlows and on the other side removes the problems connected with stability of solution for ill-conditioned systems.

The algorithm is of an iterative nature, which increases time requirements to find the solution. The calculation of products of polynomials in Algorithm II could be speeded up using methods of fast error-free convolution [1] or carrying out, when possible, the operations in parallel, e.g., in processors with vector operations.

REFERENCES

1. H.J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*, Springer-Verlag, Berlin, (1981).
2. M. Morháč, Precise deconvolution using the Fermat number transform, *Computers Math. Applic.* **12A** (3), 319–329 (1986).
3. M. Morháč, K-dimensional error-free deconvolution using the Fermat number trasform, *Computers Math. Applic.* **18** (12), 1023–1032 (1989).
4. M. Morháč, Precise multidimensional deconvolution using the polynomial algebra concept, *Intern. J. Computer Math.* **32**, 13–26 (1990).
5. M. Morháč, Determination of inverse Volterra kernels in nonlinear discrete systems. Nonlinear Analysis, Theory, *Methods & Applications* **15**, 269–281 (1990).
6. M. Newman, Solving equations exactly, *National Bureau of Standards* **71B**, 171–179 (1967).
7. R.T. Gregory, E.V. Krishnamurthy, *Methods and Applications of Error-free Computation*, Springer-Verlag, New York, (1984).
8. M. Morháč, System Identification and Deconvolution Using the Fourier and the Fermat Transforms (in Slovak), Dissertation, Bratislava, (1983).
9. M. Morháč, One modulus residue aithmetic algorithm for solving linear equations exactly (to be published).