

LUND UNIVERSITY

A Spectral Factorization Algorithm

Åström, Karl Johan

1973

Document Version: Publisher's PDF, also known as Version of record

Link to publication

Citation for published version (APA): Åström, K. J. (1973). *A Spectral Factorization Algorithm*. (Research Reports TFRT-3104). Department of Automatic Control, Lund Institute of Technology (LTH).

Total number of authors:

General rights

Unless other specific re-use rights are stated the following general rights apply:

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights. • Users may download and print one copy of any publication from the public portal for the purpose of private study or recorder.

- or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
 You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: https://creativecommons.org/licenses/

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

LUND UNIVERSITY

PO Box 117 221 00 Lund +46 46-222 00 00

A Spectral Factorization Algorithm

Ъy

K.J. Aström

Abstract.

Let B be a polynomial with zeros inside and outside the unit circlet B^* denote the reciprocial polynomial. The spectral factorizat problem is to find a polynomial C with zeros inside the unit circles uch that $BB^* = CC^*$. Some numerical algorithms to solve this problem are discussed.

Contents

1	è	INTRODUCTION

2. PRELIMINARIES

3. THE ALCORITHM

4. INTERPRETATION

5. IMPLEMENTATION

6. TEST EXAMPLES

1. INTRODUCTION

Let B be a polynomial

$$B(z) = z^{n} + b_{1} z^{n-1} + \dots + b_{n}$$

and B its reciprocial i.e.

$$B^{*}(z) = b_{n}z^{n} + b_{n-1}z^{n-1} + \dots + 1$$

The problem is to find a polynomial C with all its zeros inside the unit circle such that

 $C(z)C^{*}(z) = B(z)B^{*}(z)$

2. PRELIMINARIES

Consider the stochastic system

$$z(t+1) = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} z(t) + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ \vdots \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix} e(t)$$

 $y(t) = z_1(t) + e(t)$

where $\{e(t), t=0, \pm 1, \pm 2, \ldots\}$ is discrete time white noise with E $Ee^{2}(t) = 1$. The output $\{y(t), t=0, \pm 1, \pm 2, \ldots\}$ has the spectral dense

$$\phi_{y} = \frac{1}{2\pi} B(z) B^{*}(z')$$
 (2)

It follows from optimal filtering theory that the process $\{y(t)\}$ also can be represented as

 $\hat{z}(t+1) = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & & & \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \hat{z}(t) + \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \\ c_n \end{pmatrix} \varepsilon(t)$ $y(t) = \hat{z}_1(t) + \varepsilon(t)$

where

$$c = [\Phi P \theta^{T} + b][\theta P \theta^{T} + 1]$$

arid

The output y of (6) also can be written as

$$y(t) = \varepsilon(t) + c_1 \varepsilon(t-1) + ... + c_n \varepsilon(t-n)$$

The spectral density of the process is thus

$$\phi_{y} = \frac{1}{2\pi} C(z) C^{*}(z) \cdot k$$
 (1)

where

$$K(1+c_1^2+c_2^2+\ldots+c_n^2) = 1 + b_1^2+\ldots+b_n^2 = 1 + p_{11}$$

The equation (8) has obviously the solution P=0 which corresponds to c = b. If the polynomial B has all its zero inside the unit circle this is also the only non-negative solution. However, if B has zeros outs the unit circle there are also other solutions. In particular the polynomial time definite solution corresponds to a stable C.

3. THE ALGORITHM

The algorithm is a staight-forward iteration of the Riccati equation (8) where the initial condition is chosen in such a way that P is p sitive definite, say P(0) = I (identity matrix).

Since Φ and θ have very special forms several simplifications can h made in the algorithm. Introduce

$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & & & & \\ p_{n1} & p_{n2} & \cdots & p_{nn} \end{bmatrix}, \phi = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & & & \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ $\theta = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}$
$\theta = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}$
Then
$0P0^{T} + 1 = 1 + P_{11}$
$\Phi P \theta^{T} = \begin{bmatrix} p_{21} \\ p_{31} \\ \vdots \\ p_{n1} \\ 0 \end{bmatrix}$
$\Phi P \Phi^{T} = \begin{bmatrix} p_{22} & p_{23} & \cdots & p_{2n} & 0 \\ p_{32} & p_{33} & \cdots & p_{3n} & 0 \\ \vdots & & & & \\ p_{n2} & p_{n3} & \cdots & p_{nn} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$
$\begin{bmatrix} \mathbf{p}_{n2} & \mathbf{p}_{n3} & \cdots & \mathbf{p}_{nn} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

Hence

$$p_{ij}(t+1) = p_{i+1,j+1}(t) + b_{i}b_{j} - \frac{(p_{i+1,1}(t)+b_{i})(p_{j+1,1}(t)+b_{j})}{1+p_{11}}$$

where

 $p_{ij}(t) = 0$ if i > n or j > n $\forall t$

and

$$c = \begin{bmatrix} p_{21} + b_1 \\ p_{31} + b_2 \\ p_{n1} + b_{n-1} \\ b_n \end{bmatrix} \cdot \frac{1}{1 + p_{11}}$$

It is clear from the construction of the algorithm that its conver will depend strongly on the properties of the polynomial B. The al will converge exponentially where the exponent is determined by th of BB^* within the unit circle. The convergence thus will be very if B has zeros close to the unit circle.

4. INTERPRETATION

It is of interest to investigate the algorithm in some more detail. First observe that (14) can be written as

$$P(t+1) = \Phi P(t) \Phi + bb^{T} - (1+p_{11}(t))C(t)C^{T}(t)$$

Since $\phi^n = 0$, we get

$$P(t+n) = \sum_{i=0}^{n-1} \phi^{i} [bb^{T} - (1+p_{11}(t+n-i-1))C(t+n-i-1)C^{T}(t+n-i-1)](\phi^{T})^{i}$$

i=0

Furthermore

$$\Phi b = \begin{bmatrix} b_2 \\ b_3 \\ \vdots \\ b_n \\ b_n \\ 0 \end{bmatrix}, \quad \Phi^2 b = \begin{bmatrix} b_3 \\ b_4 \\ \vdots \\ b_n \\ 0 \\ 0 \end{bmatrix}, \dots, \quad \Phi^{n-1} b = \begin{bmatrix} b_n \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Similar equations also hold for øⁱc. Hence

$$(\phi^{i}c)_{k} = \begin{cases} c_{i+k} & \text{if} & i+k \leq n \\ & \ddots & \\ 0 & \text{if} & i+k > n \end{cases}$$

Writing the matrix equation (17) componentwise we get

$$\tilde{p}_{11}(t+n) = \sum_{i=0}^{n-1} \sum_{i=1}^{n-1} \sum_{i=0}^{n-1} \sum_{i=1}^{n-1} \sum_{i=1}^{n}$$

$$p_{21}(t+n) = \sum_{i=0}^{n-2} b_{i+2} - \sum_{i=0}^{n-2} [1+p_{11}(t+n-i-1)] c_{i+1}(t+n-i-1)c_{i+2}(t+n-i-1)c_{i$$

$$\begin{array}{c} n-1 & n-1 \\ = \sum b_{1}b_{1+1} - \sum \left[1+p_{11}(t+n-1)\right] c_{1}(t+n-1)c_{1+1}(t+n-1) \\ i=1 & i=1 \end{array}$$

$$p_{n1}(t+n) = b_1 b_n - [1+p_{11}(t+n-1)]c_n(t+n-1)c_1(t+n-1)$$

Hence

$$p_{11}(t+n) = \sum_{i=1}^{n} tb_{i}^{2} - \sigma^{2}(t+n-1)c_{i}^{2}(t+n-i)]$$

$$p_{21}(t+n) = \sum_{i=1}^{n-1} [b_i b_{i+1} - \sigma^2(t+n-1)c_i(t+n-1)c_{i+1}(t+n-1)]$$

$$P_{n1}(t+n) = b_1 b_n - \sigma^2(t+n-1)c_1(t+n-1)c_n(t+n-1)$$

where

$$\sigma^{2}(t) = [1+p_{11}(t)]$$

Furthermore introduce

$$e_{i}(t) = \frac{p_{i+11}(t) + b_{i}}{1 + p_{11}(t)} = \frac{p_{i+11}(t) + b_{i}}{\sigma^{2}(t)}$$

The above equations then can be written as

$$\sigma^{2}(t+n) = 1 + \sum_{i=1}^{n} [b_{i}^{2} - \sigma^{2}(t+n-i)c_{i}^{2}(t+n-i)] =$$

$$= \sum_{i=0}^{n} b_i^2 - \sum_{i=1}^{n} \sigma^2 (t+n-i) c_i^2 (t+n-i)$$

where

 $b_0 = 1$

Furthermore

 $c_{i}(t+n) = \frac{P_{i+1}(t+n)+b_{i}}{\sigma^{2}(t+n)} = \frac{P_{i+1}(t+n)+b_{0}b_{i}}{\sigma^{2}(t+n)} =$ $= \int_{j=0}^{n-i} b_{j}b_{j+i} - \int_{j=1}^{n-i} \sigma^{2}(t+n-j)c_{j}(t+n-j)d_{j+i}(t+n-j)]/\sigma^{2}(t+n)$

The algorithm thus can be written as follows

$$\sigma^{2}(t+n) = \sum_{i=0}^{n} b_{i}^{2} - \sum_{i=1}^{n} \sigma^{2}(t+n-i)c_{i}^{2}(t+n-i)$$

$$c_{1}(t+n) = \left[\sum_{i=0}^{n-1} b_{i}b_{i+1} - \sum_{\sigma} \sigma^{2}(t+n-j)c_{i}(t+n-j)c_{i+1}(t+n-j)/\sigma^{2}(t+n)\right]$$

$$c_{n-1}(t+n) = [b_0b_{n-1}+b_1b_n-\sigma^2(t+n-1)c_1(t+n-1)c_n(t+n-1)]/\sigma^2(t+n)$$

$$c_n(t+n) = b_o b_n / \sigma^2(t+n)$$

The algorithm discussed thus can be interpreted as a modified subs algorithm for solving the algebraic equation

$$\sigma^{2}(1+c_{1}^{2}+c_{2}^{2}+\ldots+c_{n}^{2}) = b_{0}^{2}+b_{1}^{2}+\ldots+b_{n}^{2}$$

$$\sigma^{2}(c_{1} + c_{1}c_{2} + \dots + c_{n-1}c_{n}) = b_{0}b_{1} + b_{1}b_{2} + \dots + b_{n}b_{n-1}$$

$$\sigma^{2}(c_{n-1} + c_{1}c_{n}) = b_{0}b_{n-1} + b_{1}b_{n}$$

$$\sigma^2 \cdot c_n = b_0 b_n$$

Notice, however, that the special algorithm implies that the solut: will converge to a solution where all the zeros of the C-polynomia are inside the unit circle. An alternative **thus** would be the follow algorithm

$$\sigma^{2}(t+1) = \sum_{i=0}^{n} b_{1}^{2} - \sum_{i=1}^{n} \sigma^{2}(t)c_{i}^{2}(t)$$

 $c_{1}(t+1) = \begin{bmatrix} n-1 \\ \Sigma & b_{i}b_{i+1} - \frac{n-1}{\Sigma} & \sigma^{2}(t)c_{i}(t)c_{i+1} \end{bmatrix} / \sigma^{2}(t+1)$

This algorithm does not require as much storage as (12). However, i is not immediate clear that it will always converge to the correct solution.

An Alternative

Another possibility to do the spectral factorization would be to solve the nonlinear equations by a Newton-Raphson algorithm. This gives

$$c(t+1) = c(t) - [f'(c(t))]^{-1}f(c(t))$$

where $f(c) = \begin{bmatrix} c_0^2 + c_1^2 + \dots + c_n^2 \\ c_0 c_1 + \dots + c_{n-1} c_n \\ \vdots \\ c_0 c_n \end{bmatrix} = \begin{bmatrix} c_0 & c_1 \dots & c_n \\ 0 & c_0 \dots & c_{n-1} \\ \vdots \\ 0 & 0 & \dots & c_n \end{bmatrix}$ $f'(c) = \begin{bmatrix} c_0 & c_1 \dots & c_n \\ 0 & c_0 \dots & c_{n-1} \\ \vdots \\ 0 & 0 & c_0 \end{bmatrix} + \begin{bmatrix} c_0 & c_1 \dots & c_{n-1} \\ c_n & c_n \end{pmatrix} = \begin{bmatrix} c_0 & c_1 \dots & c_{n-1} \\ c_n & c_n \end{bmatrix}$

This algorithm most likely wild converge faster than the other algorithms. However, it is again not obvious that it will converge to the correct result.

5. IMPLEMENTATION

The algorithm has been implemented in FORTRAN on PDP 15/35 as a program called SPFZN (Spectral FactoriZatioN). A test program TSPFZN and an interactive users program USPFZN also are available. These programs are listed below. A printout of the test routine also is provided.

PRINTOUT FROM TEST PROGRAM TSPFZN

TEST OF SPF2N THE ORIGINAL POLYNOHIAL IS			
1.00000000 -2.50000000 -1.00000000 -5.00000000 The Factored Polynohial is	-1.009999999	-2.47500002	0.989999999
2.20061284 -2.19992346 -3.43050772 3.97954106 THE EXACT VALUE OF FACTORED POLYNOMIAL IS	0.80510892	-1.79964948	0,44987468
2.19999999 -2.19999999 -3.43000001 3.98000002 NUMBER OF ITERATIONS = 100	0.80500001	-1.80000001	0.45000000
COMPUTING TIME 8+0 SFC		· .	
	2.78173304		
-6.54921949 2.70506752 13.20107508 -5.94561279	-6.57609677	3,15991336	· •
2.78173304 -0.53823970 -6.57609689 2.50756092	3.66526520	-1.64079985	
-1.48533177 0.55312569 3.15991336 -1.36199445 R-MATRIX	• •	0.77775559	
-0.00102502 0.00091904 0.00075185 -0.00043500 0.00091869 -0.00030309 -0.00119376 0.00042093	0.00033820	-0.00013497	
-0.00083494 0.00042105 0.00093853 -0.00047457	-0.00055242	0.00009057	
0.00013208 0.00033820 -0.00055254 -0.00017893 0.00002337 -0.00013497 0.00009674 0.00009057	0.00038302	-0.00010294 0.00002444	

LISTING OF SUBROUTINE SPFZN

001		PUDDAUT UP PDF7640 A AA Y FAA WAA
	*	SUBROUTINE SPFZN(B,C,CO,N,EPS,IND)
002	C	THIS SUBROUTINE DETERMINES & POLYNOMIAL
003	C	C(2)= -CO*(2**N+C(1)*2**(N-1)++C(N))
004	C	WITH ZEROES INSIDE THE UNIT CIRCLE SUCH THAT
005	С	$C(2)*C^{n}(2)*B(2)*B^{n}(2)$
005	C	WHERE AND A DESCRIPTION OF A DESCRIPTION
007	¢	B(Z)=Ze=N+B(1)*Z*#(N+1)+,+B(N)
008	С	
009	С	AND B" DENOTES THE RECIPROCIAL POLYNONIAL
010	Ċ	
011	Ċ	REFERENCE K.J. ASTRON "A SPECTRAL FACTORIZATION ALGORITHM"
012	С	· · · · · · · · · · · · · · · · · · ·
013	C	AUTHOR K.J. ASTROM 1971-12-29
014	č	having it as volume that the same
015	C	B - VECTOR CONTAINING COEFFICIENTS OF POLYNOHIAL
016	С	TO DE FACTORED. IT IS ASSUMED THAT LEADING
017	ĉ.	COEFFICIENT OF B 15 1.
018	С	C - VECTOR CONTAINING COEFFICIENTS OF THE FACTORED
819	č	o vertice domain and over the tents of the factored
		POLYNOMIAL AS GIVEN ABOVE. NOTICE THAT
020	C	THE COEFFICIENTS ARE NORMALIZED BY CO
021	C	CO - COEFFICIENT OF LEADING TERM OF POLYNOMIAL C
022	С	N - DEGREE OF POLYNOMIALS B AND C (MAX 15).
023	Ċ .	EDG TECT DUMITING TO TABLE DAUG CIARA 127
		EPS- TEST QUANTITY TO STOP ITERATION OF RICCATI EQUATION
024	C	IND- INDICATOR RETURNED AS 1 IF THE ITERATION.
025	C	DOES NOT CONVERGE RETURNED AS -NLOOP OTHERWISE
026	C	
027	č	SUBROUTINES REQUIRED
028	ç	NORM
029	Ċ,	
030		DIMENSION B(1),C(1)
031		CONMON/SLASK/P(16,16),R(16,16)
032	<u>^</u>	COMMAN SERVIT (10110/18/18/16)
	C	
833		IS=16
834 .		NLOP≈1000
035		NP1=N+1
036		
		DO 10 1=1,NP1
037		DO 10 J=1,NP1 .
038		R(1,J)=0.0
039	10	P(1,J)=0.0
040	~ *	
		DO 11 I=1,N
041	11	.P(,)=1.
042		NLOOP=0.
043	C	
044	Ċ	MAIN LOOP COMPUTE SOLUTION OF RICCATI EQUATION
045	č	WATH 2001 BUILOTE JOLDITOR OF REGALL EQUALIDA
046	20	R1=1,/(1,+P(1,1))
047		NLOOP=NLOOP+1 .
048		DO 21 1=1.N
049		D0 21 J=1,N
	64	
050	21	· R(1,J)≈P(1,J)
051		DO 22 I=1.N
052		DO 22 J=1,N
053 -	22	P(1,J)=B(1)*B(J)+R(1+1,J+1)~(R(1+1,1)+B(1))*(R(J+1,1)*B(J))*R[
054	F	Long and the main and the strate of the state of the stat
		DO 25 1=1,N
055	•	DD 25 J*1,N
056	25	<pre>(1, J) ~ R(1, J)</pre>
057	· C	
059	ć	TEST FOR STEADY STATE
059	č	TEST FUR STERDI STATE
	L.	
060		CALL NORH(R,N,IS,RNORM)
061		CALL NORM(P,N, IS, PHORM)
062		IF (RNORN-EPS+PNORH) 24,24,23
063	24	INDA-NLOOP
	s, ~ĭ	
- 064		60 10 29
065	23	IF(NLOOP-NLOP) 20,20,20
066	28	IND=1
067	ĉ	· · · · ·
068		POHOUTE A
	ç	COMPUTE C
069	Ċ	
070	29	T=1.+P(1,1)
071		RI=1./T
072		
	70	00 30 1=1,N
073	30	C(1)=(P(1+1,1)+B(1))+R+
074		CONSERTION

1

LISTING OF TESTPROGRAM TSPFZN

001	C	THIS IS A TEST PROGRAM FOR SPF2N
002		DIMENSION B(16), C(16), D(16)
003		COMHDN/SLASK/P(16,16),R(16,16)
004	C	
005		WRITE (6.100)
006	100	FORMAT (14H TEST OF SPF2N)
007		N÷6 -
008		NP1=#+1
009		R0=1.
010		月(1)=~2.5
011		8(2)=+1.
012		8(3)=5.
013		R(4) = -1.01
014		8(5)=-2.475
015		8(6)=,99
010	c	
017	L.	D(1)=2.2
017 018		0(2)=-2.2
		0(3)=-3.43
019		$p(4) \approx 3.98$
020		0(5)=0.805
021		B(6)=-1.8
022		8(7)=0.45
023	<u>_</u>	()(/)~(+4)
024	C	C0C+4 C-7
025		EPS=1.E-3
026		DO 50 1=1,3
027		EPS=EPS/10.
828		10F=0
029		CALL TIME10(1H, IS, IS10, IOF) CALL SPEZN (8, C, CO, N, EPS, IND)
030		
031		10F=1
032		[S1=60*1M+1S
033		TIM=FLOAT(151)+0.1+FLOAT(1510)
034		RRITE (6,103)
035	103	FORMAT (27H THE ORIGINAL PULYNONIAL IS)
036		NR(1E (6,104) H0,(B(+),(=1,#)
037	104	FORMAL (10F12.8)
038		WR(TE (6:105)
039	105	FORMAT (27H THE FACTORED POLYNOMIAL IS)
\$40		DO 20 1=1,N
041	20	C({})≠C({})≉C0
042		WRILE (6,104) CO,(C(1),1=1,N)
043		NLOP=-IND
044	106	FORMAT (23H NUMBER OF ITERATIONS =, 15)
045		WRITE (6,108)
045		WRITE (6,104) (D(1),1=1,NP1)
047		HRITE (6,106) NLOP
048		WRITE (6,102) TH
049	102	· FORMAT (15H COMPUTING TIME, F12, 1, 4H SEC)
050	108	FORMAT (42H THE EXACT VALUE OF FACTORED POLYNOMIAL 15)
051		WRITE (6,109)
052	109	FORMAT (9H P-MATRIX)
053		DO 30 1=1,N
054	30	' WRITE (6,104) (P(1,J), Jx1,N)
055		WRITE (6:110)
056	110	FORHAT (9H R+HATRIX)
057		DO 32 1=1,H
058	32	HRITE (6:104) (R(1,J), J=1,N)
059	50	CONTINUE
060		END

LISTING OF INTERACTIVE USER PROGRAM USPFZN

001	C	THIS IS A PROGRAM FOR USE OF SPF2N
002	Ē	THE PROGRAM REQUESTS THE POLYNOMIAL B FROM THE TTY
003	č	IT THEN CALLS SPFZN AND COMPUTES THE FACTORED POLYNOMIAL
004	ě.	DIMENSION B(16); C(16)
905		COMMON /SLASK/P(16,16),R(16,16)
005		DATA A/1HY/
007		
008	1	WRITE (9,100)
009	100	FORMAT (12HOTYPE DEGREF)
010	-	IONTRL=1
011		N=RTTFF(lCNTRL)
012		WR11E (9,101)
013	101	FORMAT (16H TYPE COEFF OF B)
014		ICNTRL=1
015		DO 10 I=1.N
016	10	B(1)=RTIFF(ICNTRL)
017		ICNIRL=1
018		WRITE (9,102)
019	102	FORMAT (9H TYPE EPS)
020		ICNIRL=1
021		EPS=RITFF(ICNTRL)
022		10F=0
023		
023		CALL TIME10(IM, IS, IS10, IOF)
		CALL SPF2N (B,C,CU,N,EPS,IND)
025		lof=1
026		151=60×1M+1S
027		TIM=FLOAT(IS1)+0.1*FLOAT(IS10)
028		WRITE (9,103)
029	103	FORMAL (27H THE ORIGINAL POLYNOMIAL IS)
030		WRITE (9,104) RD, (B(1), I=1,N)
031	104	FORMAT (9F8.4)
032		WRITE (9,105)
033	105	FORMAT (27H THE FACTORED POLYNOMIAL IS)
034		DO 20 1=1.N
035	20	C(1)=C(1)=C0
030		WRITE (9,104) CO.(C(I), I=1.N)
037		NLOP=-IND
038		WRITE (9,106) NLOP
039	106	FORMAT (23H NUMBER OF ITERATIONS =, 15)
040		WRITE (9,120) TIM
041	120	FORMAT (15H COMPUTING TIME, F12, 1, 4H SEC)
042		WRITE (9,107)
043	. 107	FORMAT (44H DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O))
044	- 491	READ (8,110) ANS
045	110	FORMAT (A5)
046	TTO	IF(ANS.NE.A) GO TO 1
047	30	DO 32 I=1.N
047		
	32	WRI(E (9,108) (P(1,J),J41,N)
049	108	FORMAT (6E12.4)
050	4	WRITE (9,109)
051	109	FORMAT (35H DO YOU HANT R-MATRIX Y(ES) OR N(O))
052	•	READ (8,110) ANS
053		IF (ANS.NE.A) GD 10 1
054		DO 34 1=1,N
059	34	WRITE (9:108) (R(1,J):J=1,N)
056		GO TO 1
Q\$7		END
	•	

6. TEST EXAMPLES

To construct test examples it is observed that if

$$B(z) = I(z-\alpha_i)\pi(z-\beta_i)$$

where $\alpha_{\mathbf{i}}$ are the zeros inside the unit circle and $\beta_{\mathbf{i}}$ the zeros outs the unit circle. Then

$$B^{*}(z) = H(\alpha_{i}z - 1)\pi(\beta_{i}z - 1)$$

and we find

$$C(z) = II(z - \alpha_i)\pi(\beta_i z - 1)$$

In particular we find that if the polynomial B has all its zeros outside the unit circle then

$$B(z) = b_0 z^n + b_1 z^{n-1} + \dots + b_n$$

implies that

$$C(z) = b_n z^n + b_1 z^{n-1} + \dots + b_0$$

The first set of examples is designed to check the accuracy of the algorithm. The following examples were selected.

Example 1

 $B(z) = (z-2)(z-0.5) = z^{2} - 2.5z + 1$ $C(z) = (z-0.5)(2z-1) = 2z^{2} - 2z + 0.5$

Example 2

$$C(z) = (z - \frac{9}{10})(\frac{11}{10} z - 1) = \frac{11}{10} z^2 - \frac{199}{100} z + \frac{9}{10}$$

Example 3

$$B(z) = (z - \frac{99}{100})(z - \frac{101}{100}) = z^2 - 22 + 0.9999$$

$$C(z) = (z - \frac{99}{100})(\frac{101}{100} z - 1) = \frac{101}{100} z^2 - \frac{19999}{10000} z + \frac{99}{100}$$

Example 4

 $B(z) = (z + 1)^{2} = z^{2} + 2z + 1$ $C(z) = (z + 1)^{2} = z^{2} + 2z + 1$

Example 5

 $B(z) = (z - 1)^2 = z^2 - 2z + 1$

 $C(z) = (z - 1)^2 = z^2 - 2z + 1$

Example 6

 $B(z) = (z + 1)^{3} = z^{3} + 3z^{2} + 3z + 1$ $C(z) = (z + 1)^{3} = z^{3} + 3z^{2} + 3z + 1$

Example 7

 $B(z) = z^{15} + 1.01$ $C(z) = 1.012^{15} + 1$

Example 8

$$B(z) = (z^{5}+1.01)(z^{5}+0.99) = z^{10} + 22^{5} + 0.9999$$

$$C(z) = (z^{5}+0.99)(1.01z^{5}+1) = 1.01z^{10} + 1.9999z^{5} + 0.99$$

$$Example 9$$

$$B(z) = (z^{3}+2)(z^{3}+0.5) = z^{6} + 2.5z^{3} + 1$$

$$C(z) = (2z^{3}+1)(z^{3}+0.5) = 2z^{6} + 2.5z^{3} + 0.5$$

$$Example 10$$

$$B(z) = (z^{2}-0.9)(z^{2}-1.1)(2-0.5)(z-2)$$

$$= z^{6}-2.5z^{5}-z^{4}+5z^{3} - 1.01z^{2} - 2.415z + 0.99$$

$$C(z) = (z^{2} - 0.9)(1.1z^{2} - 1)(z - 0.5)(2z - 1)$$

$$= 2.2z^{6} - 2.2z^{5} - 3.43z^{4} + 3.98z^{3} - 0.805z^{2} - 1.8z + 0.45$$

The examples were calculated using the program USPFZN. The result are given below.

TYPE DEGREE # 2 EXAMPLE 1 TYPE COEFF OF B #-2.5 1. TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 1.0000 -2.5000 1.0000 THE FACTORED POLYNOMIAL IS 2,0000 -2,0000 0.5000 NUMBER OF ITERATIONS = 11 COMPUTING TIME Ø.1 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) -Y 0.3090E+01 -0.1500E+01 -0.1500E+01 0.7500E+00 DO YOU WANT R-MATRIX Y(ES) OR N(O) Y -0.12438-03 0.5633E-04 0.5633E-04 -0.2554E-04 TYPE DEGREE Į. 2 EXAMPLE 2 TYPE COEFF OF B #-2, 6.99 TYPE EPS · #1 5-4 THE ORIGINAL POLYNOMIAL IS 1.0000 -2.0000 0.9900 - THE FACTORED POLYNOMIAL IS 1.1001 -1.9920 0.9000 NUMBER OF ITERATIONS = 61 COMPUTING TIME 0.6 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) 8 TYPE DEGREE # 2 EXAMPLE 3 TYPE COEFF OF B #-2. 0.9999 TYPD EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS. 1.0000 -2.0000 0.9999 THE FACTORED POLYNOMIAL IS 1.0105 -1.9999 0.9895 NUMBER OF ITERATIONS = 397 COMPUTING TIME 3.9 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) N

TYPE DEGREE # 2 EXAMPLE 3 WITH SMALLER EPS=1.E-8 TYPE COEFF OF B #-2. 0.9999 TYPE EPS #1.E-7 THE ORIGINAL POLYNOMIAL IS . 1.8000 -2.0800 0.9999 THE FACTORED POLYNOMIAL IS 1.0110 -1.9999 0.9890 ~ [NUNBER OF ITERATIONS = COMPUTING TIME 9.9 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) Y 0.2216E-01 -0.2191E-01 -0.2191E-01, 0.2167E-01 DO YOU WANT R-MATRIX Y(ES) OF N(O) Y Ø.8904E-05 -0.8689E-05 -0.8731E-05 0.8507E-05 TYPE DEGREE # 2 EXAMPLE 4 TYPE COEFF OF B #2. 1. TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 1.3000 2.0000 1.0000 THE FACTORED POLYNOMIAL IS 2.0000 0.9943 1.0058 NUMBER OF ITERATIONS = COMPUTING TIME 462 4.5 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) Ņ TYPE DEGREE # 2 EXAMPLE 4 WITH SMALLER EPS=1.E-7 TYPE COEFF OF B #2. 1. TYPE EPS #1.E-7 THE ORIGINAL POLYNOMIAL IS 00000 2.0000 1.0000 THE FACTORED POLYNOMIAL IS 2.0000 0.9937 1.0064 NUMBÉR OF ITERATIONS = ~] 9.9 SEC COMPUTING TIME DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(-O)

```
TYPE DEGREE
提
2 EXAMPLE 5
TYPE COEFF OF B
#-2. 1.
TYPE EPS
#1.E-4
THE ORIGINAL POLYNOMIAL IS
 1.0000 -2.0030 1.0020
THE FACTORED POLYNOMIAL IS
 1,0058 -2.0300 0.9943
NUMBER OF ITERATIONS = 463
                       4,5 SEC
COMPUTING TIME
DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
Ы
TYPE DEGREE
¥
3 EXAMPLE 6
TYPE COEFF OF P
#3. 3. 1.
TYPE EPS
#1.E-4
THE ORIGINAL FOLYNOMIAL IS
 1.0000 3.0000 3.0000 1.0000
THE FACTORED POLYNCHIAL IS
 1.0536 3.0509
                 2,9464 8,9491
NUMBER OF ITERATIONS = 724
COMPUTING TIME
                      15.2 SEC
DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
Ν
TYPE DEGREE
#
15 EXAMPLE 7
TYPE COEFF OF B
IYPE EPS
#1.E-4
 THE ORIGINAL POLYNOMIAL IS
                                                         0.0000
                         0.0000
                                         0.0000
                                                 0,0000
                                 0.0000
 1.0000 0.0000
                 0.0000
                                         0.0000
                                                 1.0100
                                 0.0000
 0.0000
         0.0000
                          0.0000
                  0.0000
 THE FACTORED POLYNOMIAL IS
                                                          0.0000
                                         0.0000
                                                 0.0000
                          0.0000
                                 0.0000
  1.0136
         0.0000
                 0.0000
                                         0.0000
                                                 0.9965
         0.0000
                                 0.0000
                          0.0000
  0.0000
                  0.0000
 NUMBER OF ITERATIONS =
                          -1
 COMPUTING TIME
                      511.1 SEC
 DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
 N
```

TYPE DEGREE # 10 EXAMPLE 8 TYPE COEFF OF B #0 0 0 0 2. 0 0 0 0 0.9999 TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 0.2000 0.0000 Ø 6.6366 6.6366 3.6369 0.0000 2.0000 1.0020 0.9999 0.0300 THE FACTORED POLYNOMIAL IS 3.0000 Ø 0.0000 1,9999 7.0360 9.0909 8.3996 0.0360 1.0129 2.9871 0.0200 NUMBER OF ITERATIONS = ~ COMPUTING TIME 227.4 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(C) Ň TYPE DEGREE # 6 EXAMPLE 9 TYPE COEFF OF B #0 0 2.5 8 0 1. TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 1.0000 0.0000 0.0000 2.5000 0.0000 0.0000 1.0000 THE FACTORED POLYNOMIAL IS 0.0000 0.0000 2.0000 0.0000 0.5000 . 0.0000 2,0000 NUMBER OF ITERATIONS = 31 2.5 SEC COMPUTING TIME DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) N TYPE DEGREE 4 6 EXAMPLE 10 TYPE COEFF OF B #-2.5 -1. 5. -1.01 -2.475 0.99 TYPE EPS #1. E-4 THE ORIGINAL POLYNOMIAL IS 1.0000 -2.5000 -1.0000 5.0000 -1.0100 -2.4750 0.9900 THE FACTORED POLYNOMIAL IS 3.9796 0.8051 -1.7997 0.4499 2.2006 -2.2000 -3.4305 NUMBER OF ITERATIONS = COMPUTING TIME 8 100 8.0 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) N

The next set of examples were selected in order to demonstrate how the computing time is influenced by the location of the zeros of the B-polynomial. Second order polynomials of the form

$$B(z) = z^2 + a^2$$
 $a > 1$

were selected. The following values of a were used a = 2 (Example 11 a = 1.5 (Example 12), a = 1.1 (Example 13), a = 1.01 (Example 14). The results are given in Table . Summarizing the number of iteration required and the computing time we get

Table 1. Illustrates the dependence of the convergence rate on the location of the zeros of the B polynomial

a	Number of iterations	Computing time
2	11	< 0.1 s
1.5	15	< 0.1 s
1.1	41	0.4 s
1.01	299	3.0 s

```
TYPE DEGREE
 H
2 EXAMPLE 11
 TYPE COEFF OF B
 #0 4
 TYPE EPS
 #1.E-4
 THE ORIGINAL POLYNOMIAL IS
  1.0000 0.0000
                 4.0.300
 THE FACTORED POLYNOMIAL IS
  4.3000 0.3030 1.0330
 NUMBER OF ITERATIONS = 11
 COMPUTING TIME
                        0.1 SEC
 DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
14
TYPE DEGREE
 #
2 EXAMPLE 12
 TYPE COEFF OF B
#0 2,25
 TYPE EPS
 #1.5-4
 THE ORIGINAL POLYNOMIAL IS
  1,3000 0,0000 2,2500
 THE FACTORED POLYHOMIAL IS
 2.2500 3.2303 1.9230
 NUMBER OF ITERATIONS =
                         15
 COMPUTING TIME
                        Ø.1 SEC
 CO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
TYPE DEGREE
 £
2 EXAMPLE 13
 TYPE COEFF OF E
 #0 1.21
 TYPE EPS
 #1.E-4
 THE ORIGINAL POLYNOMIAL IS
  1.9020 0.0000 1.2120
 THE FACTORED POLYNOMIAL IS
          8.0000 1.0020
 1.2101
 NUMBER OF ITERATIONS = 41
 COMPUTING TIME
                        2.4 SEC
 DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
N
 TYPE DEGREE
 #
2 EXAMPLE 14
 TYPE COEFF OF E
 #0 1.0201
 TYPE EPS
 #1.E-4
 THE ORIGINAL POLYNOMIAL IS
```

1.0000 0.0000 1.0201

The next set of examples (Example 15 - Example 18) are designed in order to show how the computing time depends on the size of the problem. The following polynomials were used

 $B(z) = z^n - 2^n$

The results are tabulated below.

Table 2. Illustrates how the computing time depends on the order of the B - polynomial.

Order of polynomial	Computing time	Example
3	0.2 s	15
5 4	0.4 s	16
5	0.9 s	17
	1,5 s	18
6		

```
TYPE DEGREE
2
3 EXAMPLE 15
TYPE COEFF OF B
10 2 8
TYPE EPS
#1.8-4
THE ORIGINAL POLYNOPIAL IS
 1.2820 0.6223 0.2033 8.0033
THE FACTORED POLYDOMIAL IS
 8.2330 8.3082 3.2080 1.0000
NUMBER OF ITERATIONS = 13
                       0.2 SEC
COMPUTING TIME
DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
N
 TYPE DEGREE
 #
¥
 4 EXAMPLE 16
 TYPE COEFF OF B
 #0 0 0 16
 TYPE EPS
 #1.2-4
 THE ORIGINAL POLYNOMIAL IS
  1.0000 0.0000 0.0000 0.0000 16.0000
 THE FACTORED POLYNOMIAL IS
                          0.0002 1.0000
 15.9999 8.0000 0.0000
 NUMBER OF ITERATIONS =
                         13
 COMPUTING TIME
                        9,4 SEC
 DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
 ł.
 TYPE DEGREE
  Įŀ.
  5 EXAMPLE 17
  TYPE COEFF OF B
  #2 0 0 0 32
  TYPE EPS
  #1.E-4
  THE ORIGINAL POLYNOMIAL IS
   1.0000 0.0000 0.0000 0.0000 0.0000 32.0000
  THE FACTORED POLYNOMIAL IS
  32.0000 0.0000 0.0000 0.0000 0.0000 1.0000
  NUMBER OF ITERATIONS = 16
  COMPUTING TIME
                         0.9 SEC
  DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O)
  N
  TYPE DEGREE
  #
  6 EXAMPLE 18
   TYPE COEFF OF B
   #0 0 0 0 0 64
   TYPE EPS
   #1。E-4
   THE ORIGINAL POLYNOMIAL IS
                                            a aaaa 64.0000
                                      9960
```

The examples 19 - 23 also are constructed in order to show how computing time depends on the order of the system. The following polynomials are used

 $B(z) = z_{.}^{2n} - 2^{n}$

v.

The results are tabulated below

Table 3. Illustrates how the computing time depends on the order of the polynomial.

Order of polynomial	Computing time	Example
2	0.1	19
2 4	6.7	20
·6	2.1	21.
8	3.6	22
10	7.0	23

TYPE DEGREE H 2 EXAMPLE 19 TYPE COEFF OF B #B 2. TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 1.0000 0000.0 2,8802 THE FACTORED POLYNOMIAL IS 1.9999 0.0000 1.0000 NUMBER OF ITERATIONS = 15 COMPUTING TIME 3.1 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) Į. TYPE DEGREE # 4 EXAMPLE 20 TYPE COEFF OF E #0 2 3 4. TYPE EPS #1.2-4 THE ORIGINAL POLYNOMIAL IS 1.0900 3.2222 0.3332 0.2223 4.3000 THE FACTORED POLYNOMIAL IS 3,2000 4.8888 0.2330 0.0200 1.6922 NUMBER OF ITERATIONS = 21 COMPUTING TIME Ø.7 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) R TYPE DEGREE H 6 EXAMPLE 21 TYPE COEFF OF B #0 0 0 0 0 3 TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 1.0000 0.0000 0.0000 0.0000 0.0000 0.0000 8.0000 THE FACTORED POLYNOMIAL IS 1.0000 8.0000 0.0000 0.0030 0.0000 0.0000 0.2230 NUMBER OF ITERATIONS = 25 COMPUTING. TIME 2.1 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) N

TYPE DEGREE # 8 EXAMPLE 22 TYPE COEFF OF B #0 C C O D O OFN 16. TYPE EPS #1. 8-4 THE ORIGINAL FOLYNOMIAL IS 1,0030 0,0327 8,0230 0,0323 0,0323 8,0820 8,0000 0.0009 1: THE FACTORED POLYNOMIAL IS 15,9999 0.2073 0.0000 3.2968 3.0200 3.3322 0,0030 6.0000 NUMBER OF ITERATIONS = 25 COMPUTING TIME 3.6 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) 日 TYPE DEGREE *ŧ* 10 EXAMPLE 23 TYPE COEFF OF E #0 0 0 0 0 0 0 0 0 32. TYPE EPS #1.E-4 THE ORIGINAL POLYNOMIAL IS 0.0000 0.0000 0.0000 0.0000 0.0000 1.0000 0.2020 0.0000 0.2020 32.0220 THE FACTORED POLYNOMIAL IS 0.0000 0.0000 0.0000 0.0000 0.0000 32,0000 2.0003 0.0003 0.0000 1.0000 NUMBER OF ITERATIONS = 31 COMPUTING TIME 7.0 SEC DO YOU WANT COVARIANCE MATRIX Y(ES) OR N(O) N

÷