FOREWORD

This report is a technical summary reporting the progress of a study conducted in the Mathematics Department and the Computer Center of Auburn University. The study is focused toward fulfillment of Contract No. DAAH01-68-C-0296 granted to Auburn University by the Army Missile Command, Huntsville, Alabama.

ACKNOWLEDGEMENT

The programming and documentation for this report were done by Mr. J. R. Sidbury.
ABSTRACT

A FORTRAN IV program which implements the Polynomial Manipulation System (PMS) is presented and described. PMS uses the Euclidean Algorithm to reduce a system of polynomials in several variables to a resultant system which can be solved sequentially as polynomials in one variable (Kronecker's method). PMS is described briefly and references are given to more complete discussions and to other pertinent literature.
# TABLE OF CONTENTS

I. INTRODUCTION ............................................. 1
II. PROGRAM DESCRIPTION ..................................... 2
III. BASIC FLOW CHARTS ....................................... 3
IV. INPUT/OUTPUT ............................................. 6
V. EFFICIENCY AND FUTURE WORK ............................. 6
VI. REFERENCES .................................................. 7

APPENDICES
A FORTRAN IV Program Listing ................................ 8
B Example ........................................................ 16
C Document Control Data - R & D, DD Form 1473 .......... 18
I. INTRODUCTION

The Polynomial Manipulation System (PMS) uses the Euclidean Algorithm for finding the eliminant and the greatest common divisor (g.c.d.) of two multi-variable polynomials. All polynomials involved are represented symbolically; PMS is a computer program whose input is the symbolic representation of two polynomials and whose output is (normally) the symbolic representation of their g.c.d. and eliminant.

The underlying theory and the application of PMS to the problem of solving systems of polynomial equations is discussed in [1], [2], and [3]. The present report describes a FORTRAN IV implementation of PMS developed on the IBM 360 Model 50 at Auburn University.

The program is described in Section II, the basic flow charts are given in Section III, Input/Output is discussed in Section IV, and efficiency of the method is discussed in Section V along with possible future work. The FORTRAN program is reproduced in Appendix A. Appendix B contains a simple example of the use of PMS for reduction of three polynomial equations in three variables to a resultant system which can be solved in sequence as polynomials in one variable.
II. PROGRAM DESCRIPTION

The PMS program is basically a main program with four subroutines, only one of which is significant. The other three subroutines are used for output, format headings on printed output and scaling of coefficients when they become large enough to possibly cause an overflow. In its present form the program is limited in that it is set up to use only 175K of IBM 360 storage. This limitation places constraints on the program which allows storage of only 50,000 polynomial entries (each term has \( n + 1 \) entries where \( n \) is the number of variables in the polynomial) which are presently set up as follows:

1) The pair of polynomials has, at most, four variables.

2) Each polynomial has at most 3160 terms.

3) The leading coefficient polynomials, to be defined below, can have, at most, 400 terms.

Minor modifications could increase the number of terms or variables or size of leading coefficients at the cost of decreasing the others or by use of a greater amount of machine storage. Still larger polynomials could be processed by use of tape, disc or other storage, but this has not been effected since such increases would only tend to accentuate certain disadvantages of the method to be discussed in Section V.

Consider the pair of polynomials \( U, T : \mathbb{F}^n_{\mathbb{R}}. \) Let \( x_1, \ldots, x_n \) denote the variables. Each of these polynomial functions can be considered as a polynomial in \( x_1 \) whose coefficients
would then be polynomial functions from $\mathbb{E}^{n-1}$ to $R$. Let $U^0$ and $T^0$ denote the polynomials in $x_2, \ldots, x_n$ which are the leading coefficients of $U$ and $T$ respectively considered as polynomials in $x_1$, and let $u$ and $t$ denote the degrees of $U$ and $T$ in $x_1$. We may assume $t \geq u$. Consider the polynomial $R$ defined by

$$R = U^0T - T^0Ux_1^{t-u}.$$  

$R$ is a polynomial in $x_1, \ldots, x_n$. Considering $R$ as a polynomial in $x_1$ with polynomial coefficients, it is seen that $\text{Degree}(R) < t$. Let $\text{Degree}(R) = r$. If $r \geq u$, let $T = R$ and repeat above procedure. If $r < u$, let $T = U$ and $U = R$ and repeat the above procedure. After a finite number of applications of this algorithm a polynomial $R$ will be found whose degree in $x_1$ is zero. Thus $R$ will be a polynomial in $x_2, x_3, \ldots, x_n$. It is easily seen that at each stage $R$ has any zeros that are common to $U$ and $T$. The $R$ which is free of $x_1$ is called the eliminant of $U$ and $T$.

### III. BASIC FLOW CHARTS

The flowcharts for output and scaling will be omitted as their detail is not significant to the main purpose of the program. The main program flow chart is given on page 4.
MAIN PROGRAM

Read $U, T$

$\downarrow$

Compute $U^0, T^0$

$\downarrow$

Print $U, T$

$\downarrow$

3

Determine which of $U$ and $T$ has greatest degree in $x_1$. If it is $T$ continue, if not interchange $U$ and $T$, $U^0$ and $T^0$.

$\downarrow$

Call RESIDUE
and Form $R$

$\downarrow$

Scale if necessary

$\downarrow$

If $R$ is free of $x_1$, print $R$ and return to beginning to read two new polynomials. If not, let $T = R$.  

1
RESIDUE SUBROUTINE

Form $U^0 T$

Form $-T^0 U x_t - u$

$R = U^0 T - T^0 U x_t - u$

$T = R$

Calculate new $T^0$

Is $t = 0$?

no

yes

Return to Continue Processing

Return. We Have Eliminant
IV. INPUT/OUTPUT

The polynomials U and T are read in separately. The first card for each polynomial contains the number of variables in the polynomial in columns 31-34 in integer format, right justified. The number of terms in the polynomial appear in columns 35-38 in integer format, right justified. Following this card the terms of the polynomial appear, one term per card. The coefficient appears in columns 1-16 in E format; following this are the exponents of the variables right justified in integer format in columns 17-21, 22-26, 27-31, etc.

The output of this program is available in any medium, although the program is currently set up for printed output only.

V. EFFICIENCY AND FUTURE WORK

The PMS program has not proved useful as a method of reducing polynomial systems of equations to a resultant system for the following reasons:

1) Storage efficiency is low. An inordinate amount of core is needed to process many simple appearing systems of equations.

2) Time efficiency is low. Extreme amounts of time are needed to solve all but the most simple problems. As few as four equations in four variables with small exponents (on the order of ten or less) take many hours of machine time to reach a solution. Simpler problems are solvable in
small amounts of time, but other methods without these disadvantages can be used to solve these systems.

3) Certain types of systems give solutions which have a low order of accuracy. Several articles, [7], [9], [10], have been published discussing this problem as well as the two above.

The basic PMS program will be examined and modified to determine if it is of value in algebraically solving simple systems of differential equations.

VI. REFERENCES

APPENDIX A

FORTRAN IV Program Listing

```
IMPLICIT INTEGER(*)
MAX = 40

DO 15 J = 1, MAX
   M = 0
   N = 0
   S = 0

   DO 10 I = 1, N
      DO 5 K = 1, M
         IF (VAR1(I, K) .GT. VAR2) 5 TO 106
            MAX = MAX - 1
            VAR1(I, K) = VAR2
         END IF

   END DO
   IF (VAR1(I, J) .EQ. VAR2) 5 TO 106
      MAX = MAX - 1
      VAR1(I, J) = VAR2
   END IF

   IF (VAR1(I, J) .LT. VAR2) 5 TO 106
      IF (VAR1(I, J) .LT. VAR2) 5 TO 106
         MAX = MAX - 1
         VAR1(I, J) = VAR2
      END IF

   END DO

10: MAX = MAX + 1
```

Appendix A contains a FORTRAN IV program listing. The code contains a nested loop structure, where the outer loop iterates over variables M and N, and the inner loop iterates over K. The program updates a variable VAR1 if it is greater than another variable VAR2, and it updates VAR1 to the value of VAR2. The MAX variable tracks the maximum value found, and it is incremented at the end of each iteration of the outer loop. The program has a conditional structure that checks the value of VAR1 against VAR2 and updates the value of VAR1 if necessary.
53  IUMAX(K-1,JU) = IR(K,J)
54  CONTINUE
55  DO 10 J = 1, NTERM1
56       IT(J) = IR(J)
57       DO10 I = 1, MAX
58  10  IU(I,J) = IR(I,J)
59  DO 9 J = 1, NTERM2.
60       IT(J) = IR(J)
61       DO9 I = 1, MAX
62       IU(I,J) = IR(I,J)
63  9  CONTINUE
64  CALL PRINT(IJ, IT(I,J))
65  IF(NVAR2 .GE. NVAR1) GO TO 1075
66  MAX = NVAR1
67  NVAR2 = NVAR2 + 1
68  DO 104 J = NVAR2, MAX
69       DO 104 K = 1, NTERM2
70       IR(J,K) = 0
71  104 CONTINUE
72  MAX = NVAR2
73  JPWRT = I
74  DO 56 J = 1, NTERM2
75       IF(JPWRT - IR(1,J)) 51, 60, 50
76  51 JPWRT = IR(1,J)
77  50 CONTINUE
78  JT = 0
79  DO 58 J = 1, NTERM2
80       IF(JPWRT - IR(1,J)) 57, 56, 56
81  56 JT = JT + 1
82       TMAX(JT) = R(J)
83       IF(ABS(TMAX(JT)) .GE. 1.E6) LM = 1
84       DO 58 K = 2, MAX
85  58 CONTINUE
86  ITMAX(K-1,JT) = IR(K,J)
87  56 CONTINUE
88  DO 80 J = 1, NTERM2
89       IT(J) = IR(J)
90       DO80 I = 1, MAX
91  80 IT(I,J) = IR(I,J)
92  IF(JPWRT - JPWRU) 70, 71, 71
93  70 NN = JPWRT
94  JPWRT = JPWRU
95  JPWRU = NN
96  MAXT = NTERM1
97  IF(NTERM2 .GT. NTERM1) MAXT = NTERM2
98  DO80 I = 1, MAX
99       TFM = U(I)
100      U(I) = T(I)
101      T(I) = TFM
102      DO80 J = 1, MAXT
103         TEMP = U(I,J)
104         U(I,J) = TEMP
105         TEMP = T(I,J)
106         T(I,J) = TEMP
107         IT(I,J) = IT(I,J)
108  80 IT(J,J) = TEMP
109  NN = JU
110  JU = JT
111  JT = NN
112  NN = NTERM1
113  NTERM1 = NTERM2
114  NTERM2 = NN
115  NN = NVAR1
NVAR1 = NVAR2
NVAR2 = NN
JJ=JU
IF(JT.GT.JJ) JJ=JT
MA = MAX-1
DO 73 J = 1, JJ
TW=UMAX(J)
UMAX(J) = TMAX(J)
TMAX(J)=TW
DO 73 K=1, MA
ITT=1UMAX(K, J)
UMAX(K, J) = ITMAX(K, J)
73 IF(ITT.EQ.1) CONTINUE
C
IF(LM.EQ.1) CALL SCALE2(NTERM1, NTERM2, MAX)
LM=0
101 CALL RESIUU(NTERM1, NTERM2, MAX, JWNRU, JPRTU, JU, JT)
IF(LM.EQ.1) CALL SCALE2(NTERM1, NTERM2, MAX)
IF(LM.EQ.1) GOTO 3001
IF(LS2.EQ.1) CALL SCALE2(NTERM1, NTERM2, MAX)
LS2=0
3001 LM=0
300 IF(LS1.EQ.0) GOTO 107
LS1=0
LM=0
301 CALL PRINT(3, JU, NTERM1, NTERM2, MAX)
WRITE(6, 10001) ITIMES
10000 FORMAT(IH, 5HSCALE, I3)
GO TO 100
1 FORMAT(30X,214)
4 FORMAT(F16.7,1015)
ENDD
SUBROUTINE FOR PRINTING THE TWO POLYNOMIALS AND THE RESIDUE

C
C SUBROUTINE PRINT (L,JU,NTERM1,NTERM2,N)
C
C*****************************************************************************
C*****************************************************************************
C IMPLICIT INTEGER*2(I-N)
C COMMON HMAX(400),LMAX(3,400),ITMAX(3,400),TMAX(400),R(3160),
NIR(4,3160),ITRACE,ITIMES,U(3160),IU(4,3160),T(3160)-TT(4,3160),
NLS?,LS?,L?
C
C*****************************************************************************
C*****************************************************************************
C IF(L.EQ.3) WRITE(6,310)
C IF(L.EQ.2) WRITE(6,311)
C IF(L.EQ.1) WRITE(6,312)
C CALL CRIT(H(31))
C N=31;R=1,;JU
C WRITE(*,1323) (K,(I(K),I=1,N))
C CONTINUE
C
C K = N-1
C CONTINUE
C RETURN
C 43 FORMAT(1X,51,F7.7,3X,10(I5,F5))
C 311 FORMAT(1X,'THE POLYNOMIAL ! IS!')
C 312 FORMAT(1X,'THE POLYNOMIAL ! IS!')
C 310 FORMAT(1X,'THE ELIMINANT IS!')
C 540 FORMAT(1X,214)
C 541 FORMAT(*,7,1015)
CEND
SUBROUTINE FOR PRINTING HEADINGS
SUBROUTINE PRINTH(K)

THIS SUBROUTINE MERELY PRINTS COLUMN HEADINGS FOR THE VARIABLES DEPENDING ON THE NUMBER OF VARIABLES. THAT IS ITS ONLY PURPOSE. IT WILL HANDLE UP TO 10 VARIABLES.

GO TO (21, 22, 23, 24, 25, 26, 27, 28, 29, 30), K
21 WRITE(6,31) RETURN
22 WRITE(6,32) RETURN
23 WRITE(6,33) RETURN
24 WRITE(6,34) RETURN
25 WRITE(6,35) RETURN
26 WRITE(6,36) RETURN
27 WRITE(6,37) RETURN
28 WRITE(6,38) RETURN
29 WRITE(6,39) RETURN
30 WRITE(6,40) RETURN
31 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1))
32 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2))
33 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3))
34 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4),1,5X,4HX(5))
35 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4)
1,5X,4HX(5),5X,4HX(6))
36 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4)
1,5X,4HX(5),5X,4HX(6),5X,4HX(7))
37 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4)
1,5X,4HX(5),5X,4HX(6),5X,4HX(7),5X,4HX(8))
38 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4)
1,5X,4HX(5),5X,4HX(6),5X,4HX(7),5X,4HX(8),5X,4HX(9))
39 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4)
1,5X,4HX(5),5X,4HX(6),5X,4HX(7),5X,4HX(8),5X,4HX(9),5X,4HX(10))
40 FORMAT(1HO,11HCOEFFICIENT,10X,4HX(1),5X,4HX(2),5X,4HX(3),5X,4HX(4)
1,5X,4HX(5),5X,4HX(6),5X,4HX(7),5X,4HX(8),5X,4HX(9),5X,4HX(10)) END
C  RESIDUE SUBROUTINE  FORS RESTNE AND SORTS TERMS
C
SUBROUTINE  RESIDUE(TU,NT,TV,JPU,JPT,JU,IT)
C******************************************************************************
C******************************************************************************
C
C  IMPLICIT  INTEGER*2(I-N)
C  COMMON  IMAX(400),ITMAX(3,400),ITTH(3,400),RT(3160),
C  NIR(4,3160),IT<ATE,ITIMES,UT(3160),TU(4,3160),TI(3160),IT(4,3160),
C  NLS?,LS1,Lm
C  DIMENSION  13(5)
C
MAKR=3160

MAXP = 420
TNGTEF  SMTCH

C******************************************************************************
C******************************************************************************
C
C
I =1
20  DO  J = 1,NUM
30  A=T(J)
  DO 20  K=1,TV
5210  IF(KI=IT(K,j))
       IF(IR(1,1),GF,JPT)GOTO20
       IF(AX(1A) = 1,56,48
46  LS2=1
     RETURF
45  DO 10  L = 1, JU
       III(1) = MAX(I)*A
       IR(1,1) = JU(1)
       DO 19  KK = 2,TV
10  IR(KK,1) = IMAX(KK-1,L) + IR(KK)
       4M=1-1
       IF(AK,F,9)GOTO20
       DOO=KK=1,MM
       IF(IR(1,1),NE,IR(1,1))GOTO20
       DO700KK=2,TV
       IF(IR(KKK,1),KE,IR(KKK,1))GOTO20
700  CONTINUE
       R(KK)=R(KK)+R(I)
       IF(AIKI(KKK),GT,1.0E-9)GOTO702
       NN=MM-1
       DO0=nilk=KK,NNM
       R(IK)=R(IK+1)
       DO800NPO=1,TV
800  IR(NPO,1)=IR(NPO,1)+KK
       I=1-1
702  I=I-1
     GOTO701
80  CONTINUE
701  IF(I<LT,MAXX)GOTO19
11  WRIT=(4,12)
12  FORMAT(14,25HTPS MANY TERMS IN RESIDUE)
      STOP
14  I = I + 1
20  CONTINUE
   DO 40  J = 1, ITU
A=T(J)
40  CONTINUE
DO I = 1, N
   T(I) = R(I)
   DO J = 1, N
      IT(L, J) = IR(L, J)
   END DO
END DO

376 NTT = 1
RETURN
END

SUBROUTINE SCALF2(NTFM1, NTERM2, NV)
IMPLICIT INTEGER*2(I-N)
COMMON UMAX(400), UMAX(3, 400), TMAX(3, 400), TMAX(400), R(3160),
      NTFM1, NTERM2, ITRTF, ITMES, U(3160), IU(4, 3160), T(3160), ITIM(4, 3160),
      NLS2, LS1,
EQUIVALENCE(NVAR1, MAX)
MAX = NV
NVAR2 = NVAR1
DO 1 I = 1, 400
   UMAX(I) = UMAX(I)/1000.
1   TMAX(I) = TMAX(I)/1000.
   T(J) = T(J)/1000.
   U(J) = U(J)/1000.
3   ITIMFS = ITIMFS + 1
RETURN
END
APPENDIX B

This appendix presents an example problem. The following system of three polynomials in three variables is reduced to a single polynomial in one variable:

\[(1) \quad x_1 + x_2 + x_3 = 0\]
\[(2) \quad x_1 + x^2 = 0\]
\[(3) \quad x_1^2 + x^2 = 0\]

First, \(x_1\) is eliminated between (1) and (2) producing the following printout which has been labeled for expository convenience:

```
COEFFICIENT   x(1)   x(2)   x(3)
0.1000000E 01   1       0       0  (1)
0.1000000E 01   0       1       0
0.1000000E 01   0       0       1

COEFFICIENT   x(1)   x(2)   x(3)
0.1000000E 01   1       0       0  (2)
0.1000000E 01   0       2       0

COEFFICIENT   x(1)   x(2)   x(3)
0.1000000E 01   0       2       0  (3)
-0.1000000E 01   0       1       0
-0.1000000E 01   0       0       1
```

Second, \(x_1\) is eliminated between (3) and (2) producing the following printout:

```
COEFFICIENT   x(1)   x(2)   x(3)
0.1000000E 01   1       0       0  (3)
0.1000000E 01   0       2       0

COEFFICIENT   x(1)   x(2)   x(3)
0.1000000E 01   1       0       0  (2)
0.1000000E 01   0       2       0

COEFFICIENT   x(1)   x(2)   x(3)
0.1000000E 01   0       2       0  (3)
-0.1000000E 01   0       0       2  (E_1)
-0.1000000E 01   0       0       1
```

15
Finally (E₁) and (E₂) are treated as a pair of polynomials in two variables x₁ and x₂. Then x₁ (x₂ in our first system) is eliminated, producing the following printout:

<table>
<thead>
<tr>
<th>COEFFICIENT</th>
<th>x(1)</th>
<th>x(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1000000E 01</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>-0.1000000E 01</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>-0.1000000E 01</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COEFFICIENT</th>
<th>x(1)</th>
<th>x(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.1000000E 01</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0.1000000E 01</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COEFFICIENT</th>
<th>x(1)</th>
<th>x(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1000000E 01</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>-0.2000000E 01</td>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

(E₃) is our eliminant free of x₁ and x₂, so it can be solved. Using its solution (E₂) can then be solved. Using this (2) can be solved and the solutions to the resultant system (2), (E₂), (E₃) are the solutions to the system (1), (2), (3). Three other equations from these six could have been taken to form the resultant system.
POLYNOMIAL MANIPULATION SYSTEM-FORTRAN IV PROGRAM

A FORTRAN IV program which implements the Polynomial Manipulation System (PMS) is presented and described. PMS uses the Euclidean Algorithm to reduce a system of polynomials in several variables to a resultant system which can be solved sequentially as polynomials in one variable (Kronecker's method). PMS is described briefly and references are given to more complete discussions and to other pertinent literature.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINE A</th>
<th>LINE B</th>
<th>LINE C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polynomials</td>
<td>ROLE XT</td>
<td>ROLE XT</td>
<td>ROLE XT</td>
</tr>
<tr>
<td>Resultant</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eliminant</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Euclid's Algorithm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORTRAN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Program</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>