ANLYZ - A DIGITAL COMPUTER PROGRAM FOR THE COMPLEX ANALYSIS OF DISTRIBUTED-LUMPED-ACTIVE NETWORKS

Prepared under Grant NGL-03-002-136 for the
Instrumentation Division of the Ames Research Center
National Aeronautics and Space Administration

by

L. P. Huelsman

ENGINEERING EXPERIMENT STATION
COLLEGE OF ENGINEERING
THE UNIVERSITY OF ARIZONA
TUCSON, ARIZONA

ANLYZ - A DIGITAL COMPUTER PROGRAM FOR THE COMPLEX ANALYSIS OF DISTRIBUTED-LUMPED-ACTIVE NETWORKS

Prepared under Grant NGL-03-002-136 for the
Instrumentation Division of the Ames Research Center
National Aeronautics and Space Administration

by

L. P. Huelsman

Department of Electrical Engineering
University of Arizona
Tucson, Arizona

Abstract: This report describes ANLYZ, a digital computer program for the complex analysis of distributed-lumped-active networks. The program is designed to be used in complex optimization studies of a broad class of this type of network. The program provides as output the magnitude (or the reciprocal of the magnitude) of the voltage transfer function for the network as evaluated at a specific point in the complex frequency plane. The program is designed to be used with the GOSPEL optimization software package described in a previous report.

October 1969

Table of Contents

ANLYZ - A Digital Computer Program for Complex Optimization of DLA Networks

I. Introduction

This is one of a series of reports concerning the use of digital computational techniques in the analysis and synthesis of DLA (distributed-lumped-active) networks. This class of networks consists of three distinct types of elements, namely distributed elements (modeled by partial differential equations), lumped elements (modeled by algebraic equations and ordinary differential equations), and active elements (modeled by algebraic equations). Such a characterization is especially applicable to the broad class of circuits referred to as linear integrated circuits, since the required fabrication techniques readily produce elements which may be referred to as "distributed", as well as producing elements which may be characterized as "lumped" and/or "active". The DLA class of networks is capable of realizing network functions with a wide range of properties. In addition, such realizations usually have fewer components and superior characteristics than realizations using only lumped elements, or realizations using lumped elements and active elements. The analysis problem for this class of networks, however, is considerably more complex than the analysis problem for more restricted classes of networks. Thus, one is lead logically to consider using the digital computer as an analysis for DLA networks. Previous reports have described a digital computer program for making such analyses.[1,2] The program is called DLANET and it has the capability for providing a sinusoidal steady state analysis of a broad class of DLA networks of prescribed frequency ranges.

A problem associated with DLA networks which is of even greater complexity than the analysis problem is the synthesis problem, i.e., determining the topology and the parameter values for a network so as to realize a specified set of characteristics. The analytical synthesis methods whcih have been discovered to date have been too specialized and too complicated to be of any appreciable value. Thus, again, the application of the digital computer appears to be called for. As a preliminary step to such an application, in a previous report a general optimization software package named GOSPEL was described.[3] This program provided a means for using the digital computer to solve a broad range of optimization problems. The program was designed so that it could be immediately applied to specific DLA network configurations, however, it was necessary that the equations describing such configurations had to be re-programmed for each different network configuration. In this report we describe a network program similar in fuction to the general DLANET program referred to above but designed so that it may be directly used with the GOSPEL optimization program. Thus, as required by GOSPEL, this network program is prepared in subroutine form and is named ANLYZ. The ANLYZ program has been specifically designed so that it will analyze a broad range of DLA networks in such a manner than any of the individual optimization strategies of the GOSPEL program may be directly applied to it. In addition, the program has been written so that an analysis may be made not only at sinusoidal frequencies but also at complex frequencies. Thus, it may be directly applied as a means to achieve complex optimization. In such a usage the values of the parameters of a given network are varied so as to minimize the magnitude of the network function at given points in the complex frequency plane where zeros of transmission are desired and also to maximize the magnitude at complex frequencies where poles are desired. An optimization strategy such as those available at GOSPEL is used to determine the proper variations of the network parameters. Thus, dominant zeros and poles are effectively created to realize some specified network characteristic. More details of the complex optimization procedure may be found in a paper which was prepared under this grant.[4] The actual operation of the ANLYZ subroutine is described in more detail in the following sections of this report. The treatment given here assumes that the reader has a knowledge of the basic operation of the DLANET and GOSPEL programs as covered in the above referenced reports.

II. General Theory of the Digital Computer Program ANLYZ

In this section we describe the general theory of the digital computer program named ANLYZ. The general operation of this program is determined by the following functions:

(1) It reads input data describing the specific DLA network configuration formed as an interconnection of any of the allowed network elements. These include lumped resistors, lumped capacitors, distributed RC networks of various tapers, and single or differential-input VCVSs.

(2) It reads input data determining which of the network parameters are to be optimized (or varied), and which are to remain fixed. For the latter, the actual parameter values are read as input data.

(3) It has provision for receiving as input the complex numbers specifying the frequencies at which dominant poles or zeros of the network function are desired. These are entered from the calling program (GOSPEL) by the use of common variables.

(4) It reads input data consisting of constants which determine whether each of the specific complex frequencies described in (3) above are to be considered as poles or zeros. This determines whether the magnitude of the network function or its reciprocal is to be minimized.

(5) It has provision for receiving as input, the current values of the variable network parameters as determined by the optimization algorithm. These are also entered into the program as common variables.

(6) It computes the magnitude of the network function at each of the specified complex frequencies provided in (3) and supplies these values (or their reciprocal) as outputs. The outputs are returned to the GOSPEL optimization program by means of common variable linkages.

In the above list, items (1), (2), and (4) provide information on the topology of the DLA network which is being analyzed and on other invariant specifications of the problem. As such these items need be implemented only once in the optimization process. Thus, they comprise a section of the ANLYZ program which is executed only once, namely, the first time the program is called. This call will normally be made as part of the operation of the GOSPEL program. Items (3) and (5) provide for input of data directly from the GOSPEL program. Such input is provided by using the same common variables in the subroutine ANLYZ as are used in GOSPEL. As described in the GOSPEL report, these variables are stored in a block of common storage labeled OPT. Finally, item (6) provides for the actual computation of the magnitude of the network function at specific frequencies. The logic for doing this is similar to that used in DLANET. A major difference, however, is the use of complex arithmetic, which makes it possible for the ANLYZ subroutine capable to be applied to complex optimization problems.

III. Details of the Operation of the Digital Computer Program ANLYZ

In this section we present a description of some of the details of the operation of the digital computer program ANLYZ. A flow chart of the program may be found in Appendix A and a listing of the Fortran statements in Appendix B.

The program ANLYZ consists of a main subroutine called ANLYZ and three accompanying subroutines named YDA, CMRD and CONS. First let us consider the subroutine ANLYZ. This has two main functions. These are reading input data not supplied as common variables from the calling program GOSPEL, and performing computations (using complex arithmetic) to determine the required magnitude of the network transfer function. The first function, i.e., the reading of input data, is performed only once, namely, the first time the subroutine is called. To accomplish this a constant KREAD is initialized to zero at the time the subroutine is compiled by the use of a data statement. At the completion of the input data reading part of the program this constant is then set to unity. On subsequent calls of the subroutine a test of the value of this constant is made. If it is unity, the data reading portion of the subroutine is bypassed. The input data reading operation begins by defining a constant NHZ which is equal to NH/2 where NH is the number of values of the functional parameters specified in GOSPEL. Since, for complex optimization, each complex frequency requires two functional parameters, NHZ is thus the number of complex frequencies at which an evaluation of the network transfer function is to be made. The NHZ constants KPZ(I) (I = 1, NHZ) next read as zeros for each frequency which is to be considered as a zero, or ones for those which are to be considered as poles. Next a series of constants specifying the number of nodes; the number of distributed networks, resistors, capacitors, and VCVSs; the number of resistors optimized, capacitors optimized, and VCVSs optimized; and the data for the distributed networks including the node numbers to which each network is connected, the number of sections to be used in the lumped model, the number of polynomial coefficients used in describing the taper, and indicators for designating whether the resistance or the capacitance or the polynomial coefficients are to be optimized. The program network now considers each type of element in turn. Using the variable IP as the index of the optimized variables stored as X(I) in common with GOSPEL it successively prints the appropriate X(I) value if a variable is to be optimized or reads and prints the data if the variable is fixed.

The second purpose of the subroutine ANLYZ is implemented by the portion of the program beginning at statement 265. This portion is repeated at each call of the subroutine. The logic begins by assigning the variables X(I) which are to be optimized in correct sequence as variables of the DLA network. This is done by using the quantity NX as an index variable. Next a DO loop is entered for the computation of the NHZ values of the transfer function magnitude. Two successive values of the functional parameters H(I), as read by GOSPEL, are assigned as the real and imaginary parts of each complex frequency. The subroutine YDA is then called to compute models for the distributed networks and to store the effects of these elements in the complex admittance array YR. The operation of this subroutine is similar to that of the subroutine YDIST in the program DLANET. Next, the values of resistance and capacitance are added to the appropriate elements of the YR array. The values of capacitance are, of course, multiplied by frequency and treated as imaginary quantities. The effect of the VCVSs on the network is now taken into consideration by appropriately constraining the YR array using the subroutine CONS. The operation of this subroutine is similar to that of the CONST subroutine in the program DLANET. Finally, the subroutine CMRD (similar to CMRED in DLANET) is used to solve for the voltage transfer function. The results are stored in the common variables G(I) directly, or after inversion, depending on whether the frequency being considered is specified as a zero or a pole. The program GOSPEL uses NH values of the quantities G(I) in computing the error function. Since there are only NH/2 determinations of the transfer function made by the subroutine ANLYZ, each value so computed is stored in two successive locations of the G(I). At the end of the last transfer function magnitude computation the subroutine ANLYZ returns control to the calling program, namely, GOSPEL. The transfer of variables between ANLYZ and its component subroutines is through the use of a common block of storage labeled DLA. The program is written in FORTRAN IV, and, although it has been developed on the CDC 6400 computer, it is applicable to a broad range of medium to large size digital computers.

IV. The Use of the Digital Computer Program ANLYZ

The digital computer program ANLYZ is designed for use in complex optimization studies of a broad class of DLA networks. As such, it is designed to be used with one or more of the optimization strategies of the GOSPEL software package. It is also readily usable for $j\omega$ axis matching of transfer function characteristics by using complex frequencies in which the real part is set to zero. As currently dimensioned, the program has the capability of analyzing networks with up to nine nodes (not including the reference node). In addition, as currently dimensioned the following upper limits apply: five distributed networks, ten resistors (of which five may be optimized), ten capacitors (of which five may be optimized) and five VCVSs. The distributed networks may be modeled as uniform elements in which the resistance and capacitance per unit length are constant, or as elements with a polynomial taper. In the latter case the variation of resistance and capacitance per unit length is given as

$$r(x) = R_o(1 + p_1x + p_2x^2 + p_3x^3 + p_4x^4 + p_5x^5)$$

$$c(x) = C_o/(1 + p_1x + p_2x^2 + p_3x^3 + p_4x^4 + p_5x^5)$$

For a uniform line the total resistance or the total capacitance or both may be optimized. For a tapered line, the quantities R_o and C_o in the above relations plus as many polynomial coefficients as desired may be optimized. A list of the order of which the optimized variables X(I) of the program GOSPEL are assigned to DLA network variables is given in Appendix C. A list of the format used for the input data cards for the program is given in Appendix D. It should be noted that, in assigning numbers to the various nodes of the network, the input node must always be labeled as node 1 and the output node as node 2.

As an example of the use of this program consider the network shown in Fig. 1. The network function which it is desired to realize has four complex-conjugate dominant poles and realizes a maximally-flat-magnitude low-pass characteristic. Thus, it has the form

$$\frac{V_2}{V_1} = \frac{H}{(p - p_1)(p - \bar{p}_1)(p - p_2)(p - \bar{p}_2)}$$

A listing of the input data and the output data which were obtained from using subroutine OPT9 (the Fletcher-Powell optimization subroutine) of the GOSPEL software package are given in Figs. 2 and 3. The starting point was selected from a set of values given for this network in the literature.[5] It is readily observed that a decrease in the total error was obtained as a result of the application of the optimization strategy. Thus, the program achieved an improvement in the dominance of the designated poles at which the behavior of the network was evaluated.

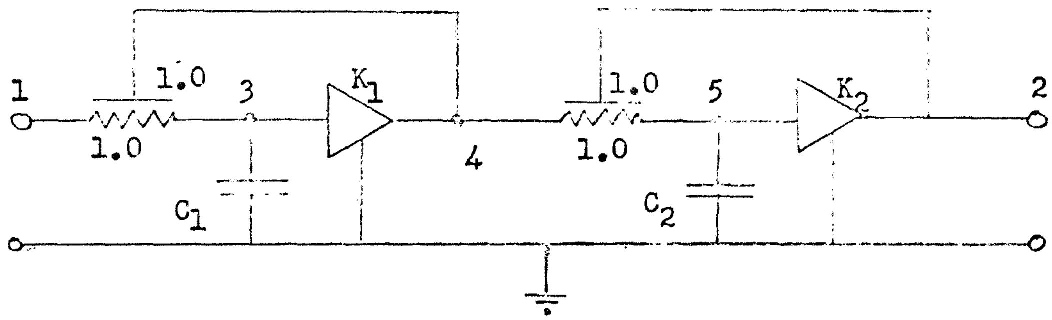


Figure 1

```
TEST OF DLA ANLYZ ON 2 BUNKER I-A STAGES, MFM, P.32,X1=C1,X2=C2,X3=K1,X4=K2
 4  4 40    1.      E-C7
 1
11.5631    11.7922    20.3007    22.9217
 7
-.38268    .92388     -.92388    .38268
 8
0.         0.         0.         0.

   1   1
   5   2   0   2   2   0   2   2
   1   3   4  50   0   0   0   0
   4   5   2  50   0   0   0   0
1.
1.
1.
1.
   3   6   5   6
   4   0   3   0   4
   2   0   5   0   5
```

Figure 2

```
TEST OF DLA ANLYZ ON 2 BUNKER I-A STAGES, MFM, P.32,X1=C1,X2=C2,X3=K1,X4=K2

MINIMUM ERROR = 1.000E-07      MAXIMUM ITERATIONS =  40

INPUT DATA    4 VARIABLES    4 DATA POINTS


INITIAL VALUES OF VARIABLES
 1.156E+01  1.179E+01  2.030E+01  2.292E+01


DATA POINTS
-3.827E-01  9.239E-01 -9.239E-01  3.827E-01


DESIRED VALUES
 0.          0.          0.          0.


OPT9 FLETCHER POWELL HAS BEEN CALLED

PARAM(9,1)-PERTURBATION SIZE FOR FINDING GRADIENT = 1.00E-06
PARAM(9,2)-MAX ITERATIONS IN ONE DIMENSIONAL SEARCH = 2.00E+01
PARAM(9,3)-NUMBER OF RESET CYCLES PERMITTED= 3.00E+00
NOPT(9,1)-FIND GRADIENT BY (0) PERTURBATION, (1) ANLYD.... 0
NOPT(9,2)-PRINTOUT,(-)NONE,(0) REDUCED, (1) 1-D, (2) ABH.. 0


FIRST CALL OF DLA ANLYZ, OPTIMIZATION AT  2 FREQUENCIES

FREQUENCY  1 IS A POLE LOCATED AT -3.83E-01+J 9.24E-01
FREQUENCY  2 IS A POLE LOCATED AT -9.24E-01+J 3.83E-01
NUMBER OF NODES (REFERENCE NODE NOT INCLUDED)..............  5

DISTRIBUTED NETWORK 1
NUMBER OF SECTIONS IN DISTRIBUTED NETWORK.................. 50
DISTRIBUTED NETWORK CONNECTED TO NODES...............  1  3  4
DISTRIBUTED R OPTIMIZED (1=YES, 0=NO).......................  0
DISTRIBUTED C OPTIMIZED (1=YES, 0=NO).......................  0
NUMBER OF TAPER COEFFICIENTS (0=UNIFORM LINE)...............  0
        FIXED DISTRIBUTED RESISTANCE............... 1.0000E+00
        FIXED DISTRIBUTED CAPACITANCE.............. 1.0000E+00

DISTRIBUTED NETWORK 2
NUMBER OF SECTIONS IN DISTRIBUTED NETWORK.................. 50
DISTRIBUTED NETWORK CONNECTED TO NODES...............  4  5  2
DISTRIBUTED R OPTIMIZED (1=YES, 0=NO).......................  0
DISTRIBUTED C OPTIMIZED (1=YES, 0=NO).......................  0
NUMBER OF TAPER COEFFICIENTS (0=UNIFORM LINE)...............  0
        FIXED DISTRIBUTED RESISTANCE............... 1.0000E+00
        FIXED DISTRIBUTED CAPACITANCE.............. 1.0000E+00

NUMBER OF LUMPED RESISTORS..................................  0
NUMBER OF LUMPED RESISTORS OPTIMIZED........................  0
NUMBER OF LUMPED CAPACITORS.................................  2
NUMBER OF LUMPED CAPACITORS OPTIMIZED.......................  2
X( 1) - OPTIMIZED LUMPED CAPACITOR  1 NODES  3  6.. 1.1563E+01
X( 2) - OPTIMIZED LUMPED CAPACITOR  2 NODES  5  6.. 1.1792E+01

X( 3) - OPTIMIZED VCVS  1, OUTPUT NODES  4  0
        INPUT NODES  3  0, NODE  4 ELIM. . . . 2.0301E+01
```

```
X( 4) - OPTIMIZED VCVS  2, OUTPUT NODES  2  0
             INPUT NODES  5  0, NODE  5 ELIM....... 2.2922E+01

ITERATION   0              ERROR= 6.646E-07
X(I)     1.156E+01  1.179E+01  2.030E+01  2.292E+01

ITERATION   1     KA=18    ERROR= 2.239E-09      ALFA= 5.036E+04
X(I)     1.172E+01  1.196E+01  2.022E+01  2.284E+01

H MATRIX
 1.00E+00  0.         0.         0.
 0.        1.00E+00   0.         0.
 0.        0.         1.00E+00   0.
 0.        0.         0.         1.00E+00


SUMMARY OF OPTIMIZATION RESULTS


   1 ITERATIONS      2.23896957E-09 FINAL ERROR


  FINAL VARIABLE VALUES

   1       1.17202883E+01
   2       1.19565366E+01
   3       2.02165177E+01
   4       2.28379889E+01

  FINAL VALUES OF G(I)

   1       2.61604596E-05
   2       2.61604596E-05
   3       2.08594137E-05
   4       2.08594137E-05


   08/28/69   SCOPE 3.1.2H    UNIVERSITY OF ARIZONA
 12.57.57.HUEL35J
 12.57.57.HUEL3,T10,CM45000,P17,BN3900082Y.
 12.57.57.RUN(S)
 12.58.04.LOAD,INPUT.
 12.58.08.LGO.
 12.58.14.FL= 045000,   CPU=  000.482,    PPU=  000.680
 12.58.17.STOP
 12.58.18.FL= 022500,   CPU=  003.169,    PPU=  000.918
```

Figure 3

V. Conclusion

In this report the theory and application of a digital computer program named ANLYZ was described. This program has the capability of performing complex optimization for a broad class of DLA networks. Thus, it provides a valuable general tool for research in this area. It should be noted that if extensive optimization studies on a specific DLA network configuration are to be carried out, it may be more effective, computationally, to prepare a program ANLYZ specifically for the network which is being treated. In such a case, the general digital computer program ANLYZ described in this report will prove extremely valuable to provide verification of the correctness of such a specific program.

Acknowledgement

The author wishes to acknowledge the support given to this research by the Instrumentation Division of the Ames Research Center, National Aeronautics and Space Administration, Moffett Field, California.

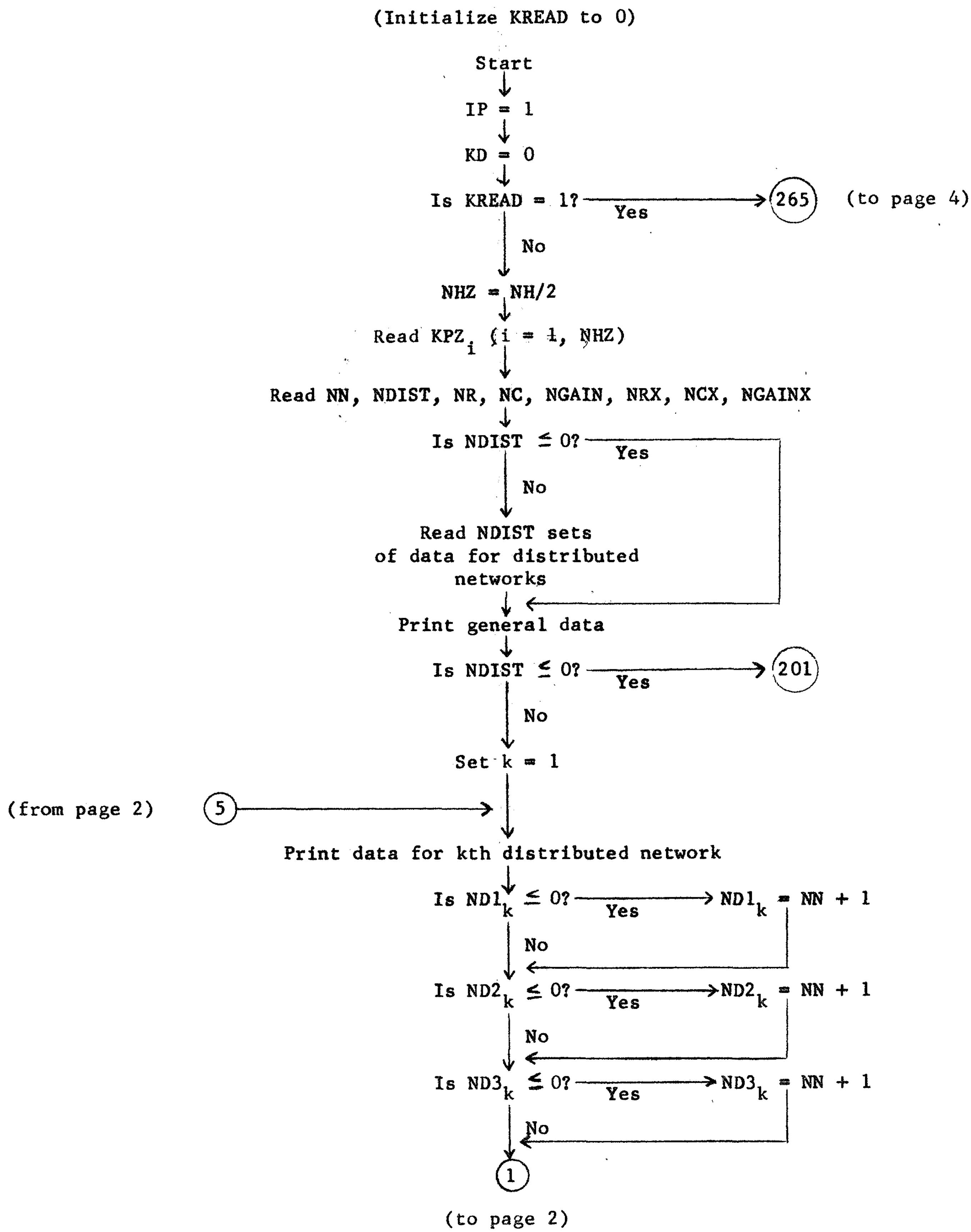
References

1. L. P. Huelsman, DLANET - A Digital Computer Program for the Analysis of Distributed-Lumped-Active Networks, Engineering Experiment Station Report, University of Arizona, 46 pages, Nov., 1969.
2. M. L. Gentry and L. P. Huelsman, Modification of the Digital Computer Program DLANET to include the Effects of Differential Input Voltage-Controlled Voltage Sources, Engineering Experiment Station Report, University of Arizona, 13 pages, Mar., 1969.
3. L. P. Huelsman, GOSPEL - A General Optimization Software Package for Electrical Network Design, Engineering Experiment Station Report, University of Arizona, 94 pages, Sept., 1968.

4. L. P. Huelsman, Synthesis of Distributed-Lumped-Active Networks by Complex Optimization, Proceedings of the Second Hawaii International Conference on System Sciences, pp. 167-171, Jan. 22-24, 1969.

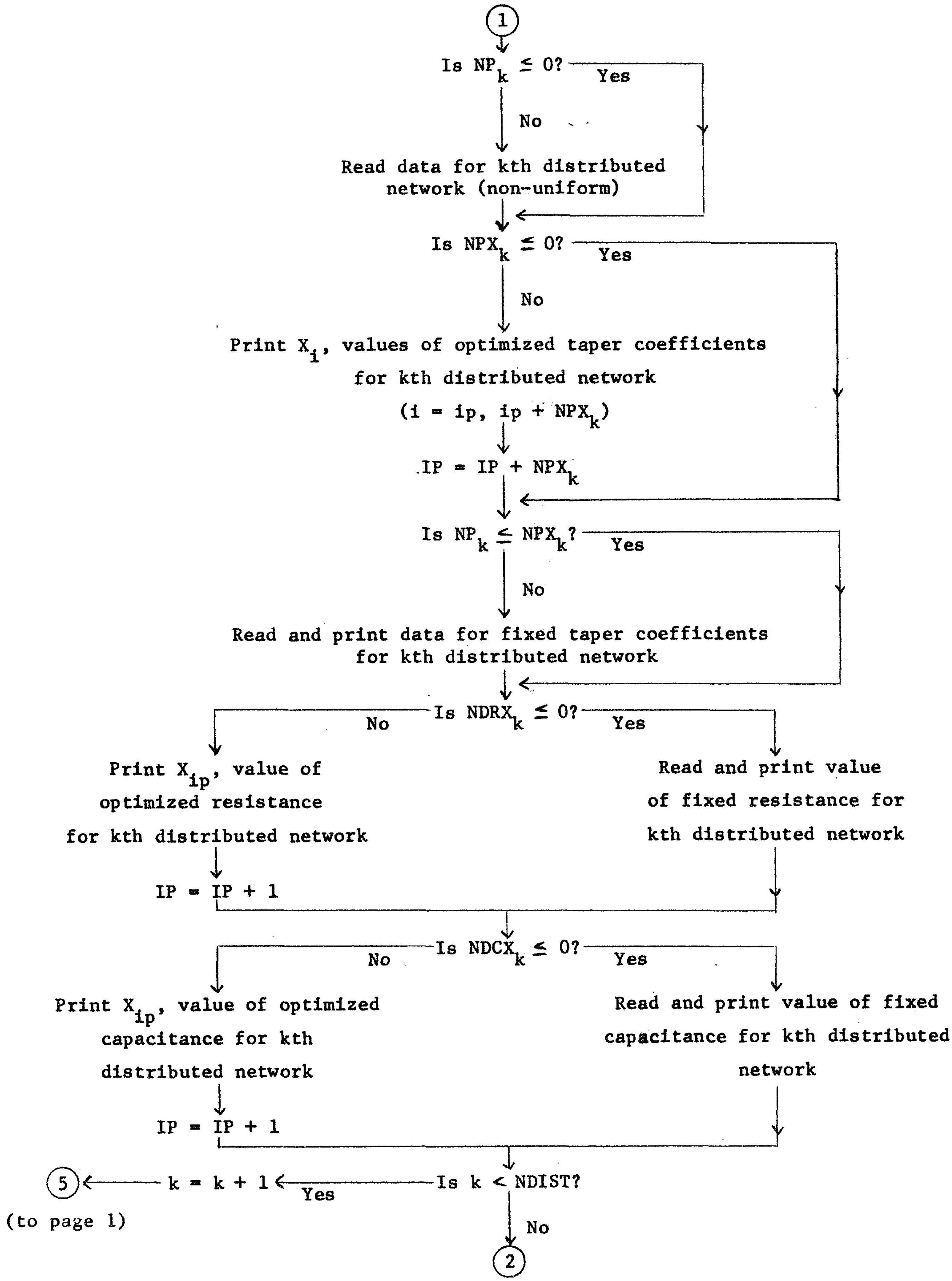
5. W. Marvin Bunker, Exact Synthesis of Multistage Active Networks Containing Distributed RC Elements, General Electric Report 68ASD30, Apollo Systems Dept., Missile and Space Division, GE Co., Daytona Beach, Florida, Dec. 31, 1968.

Flow Chart for ANLYZ Subroutine for DLA Network



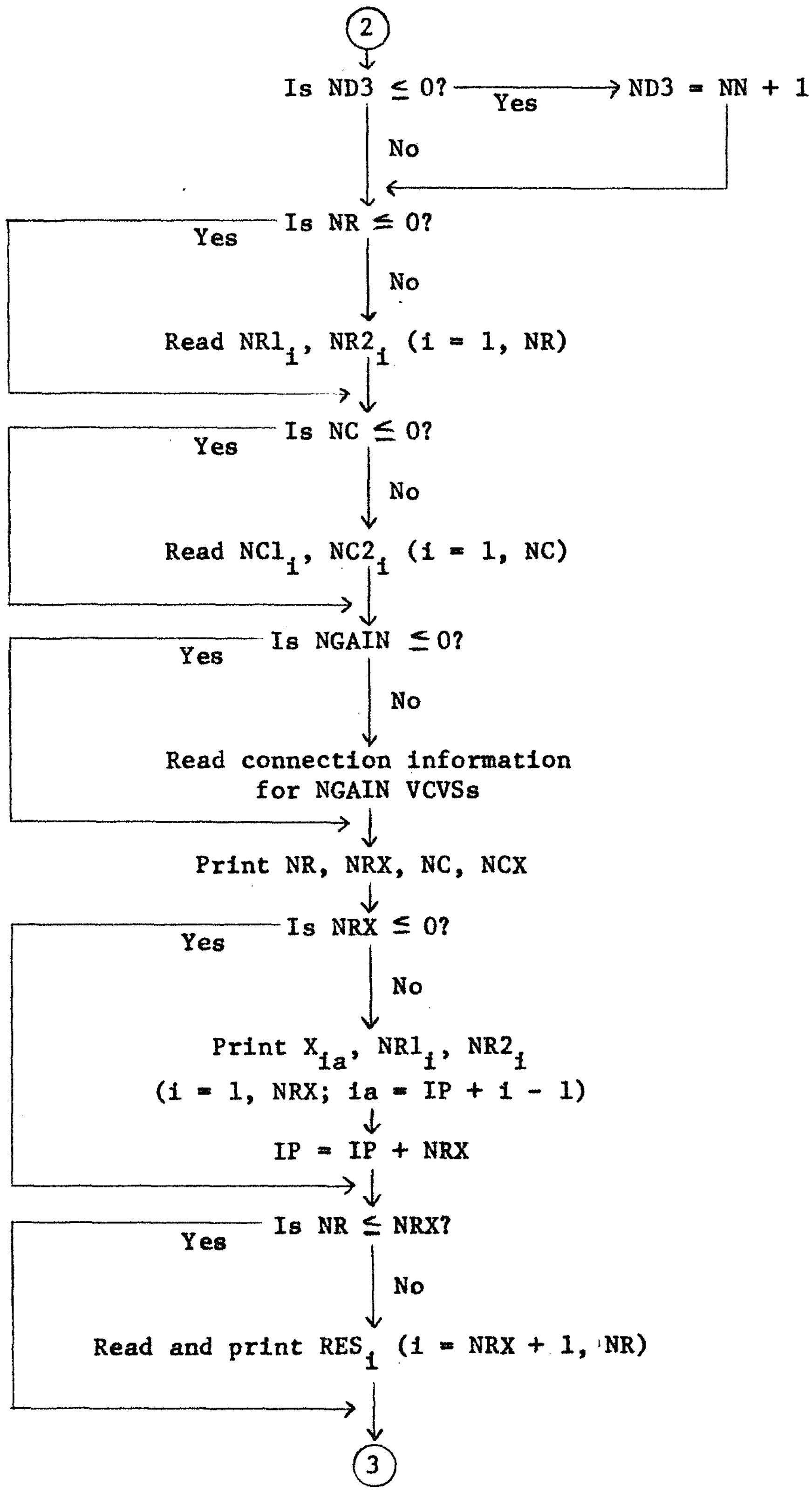
Flow Chart Page 2

(from page 1)



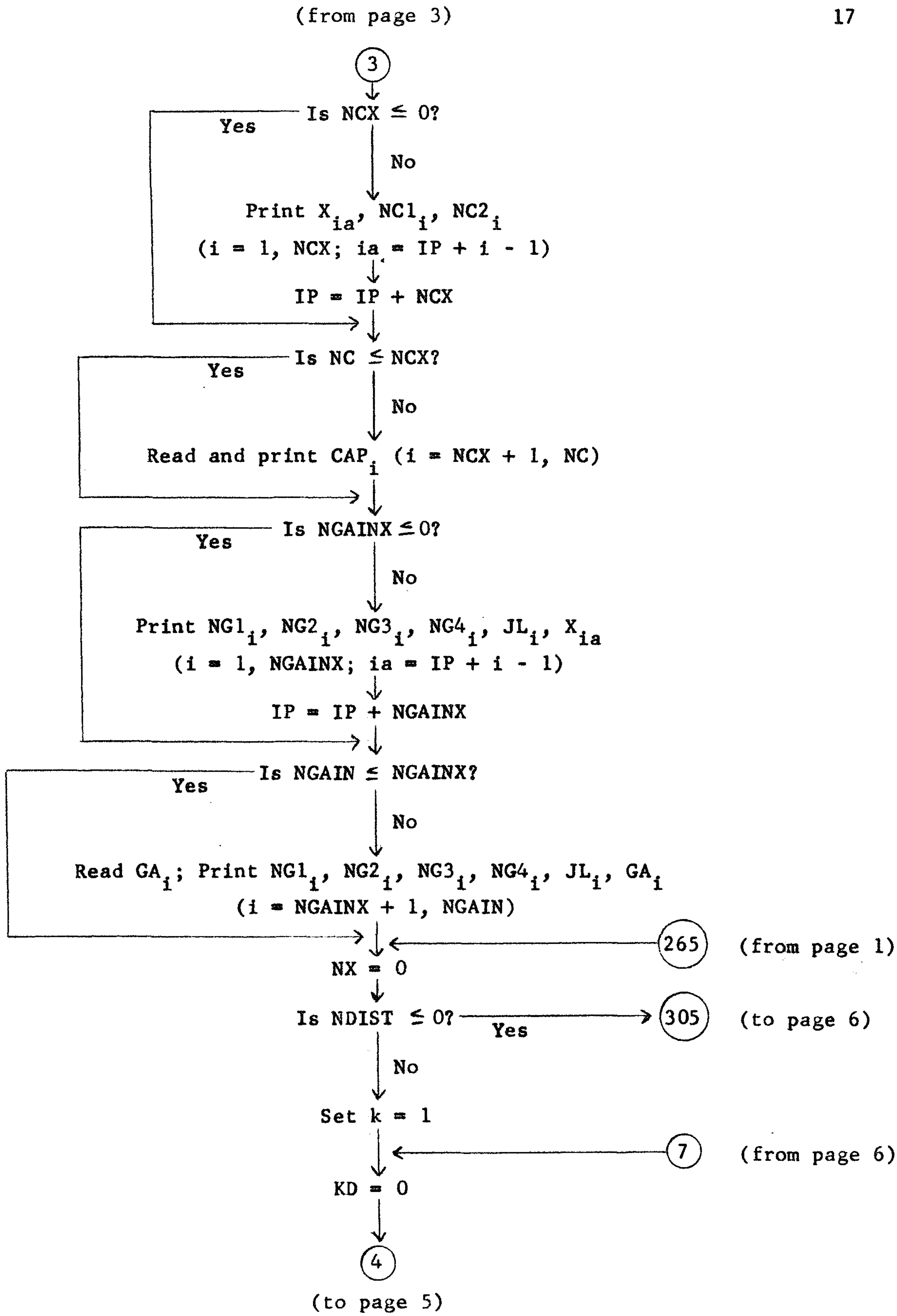
(to page 3)

(from page 2)

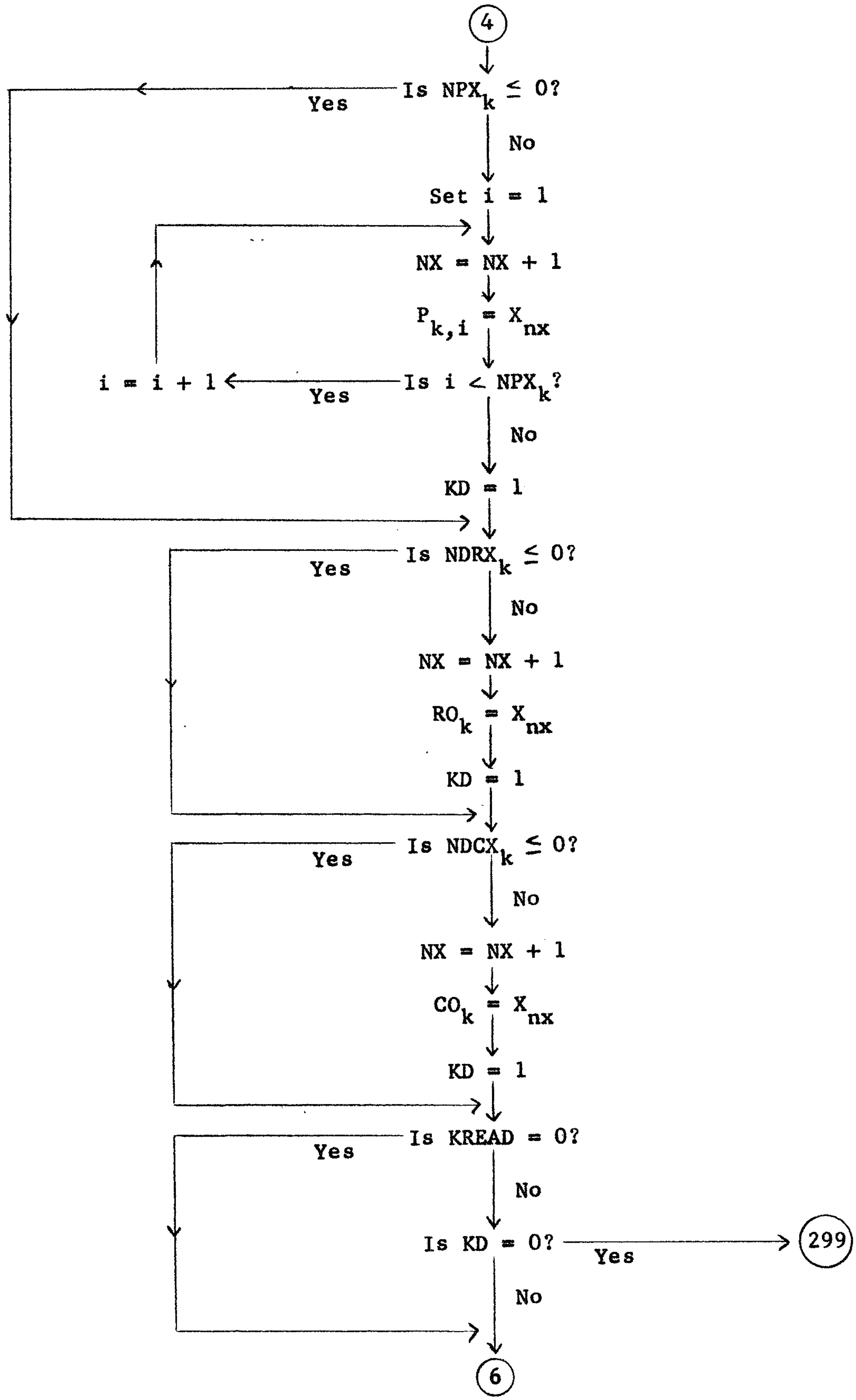


(to page 4)

Flow Chart Page 4



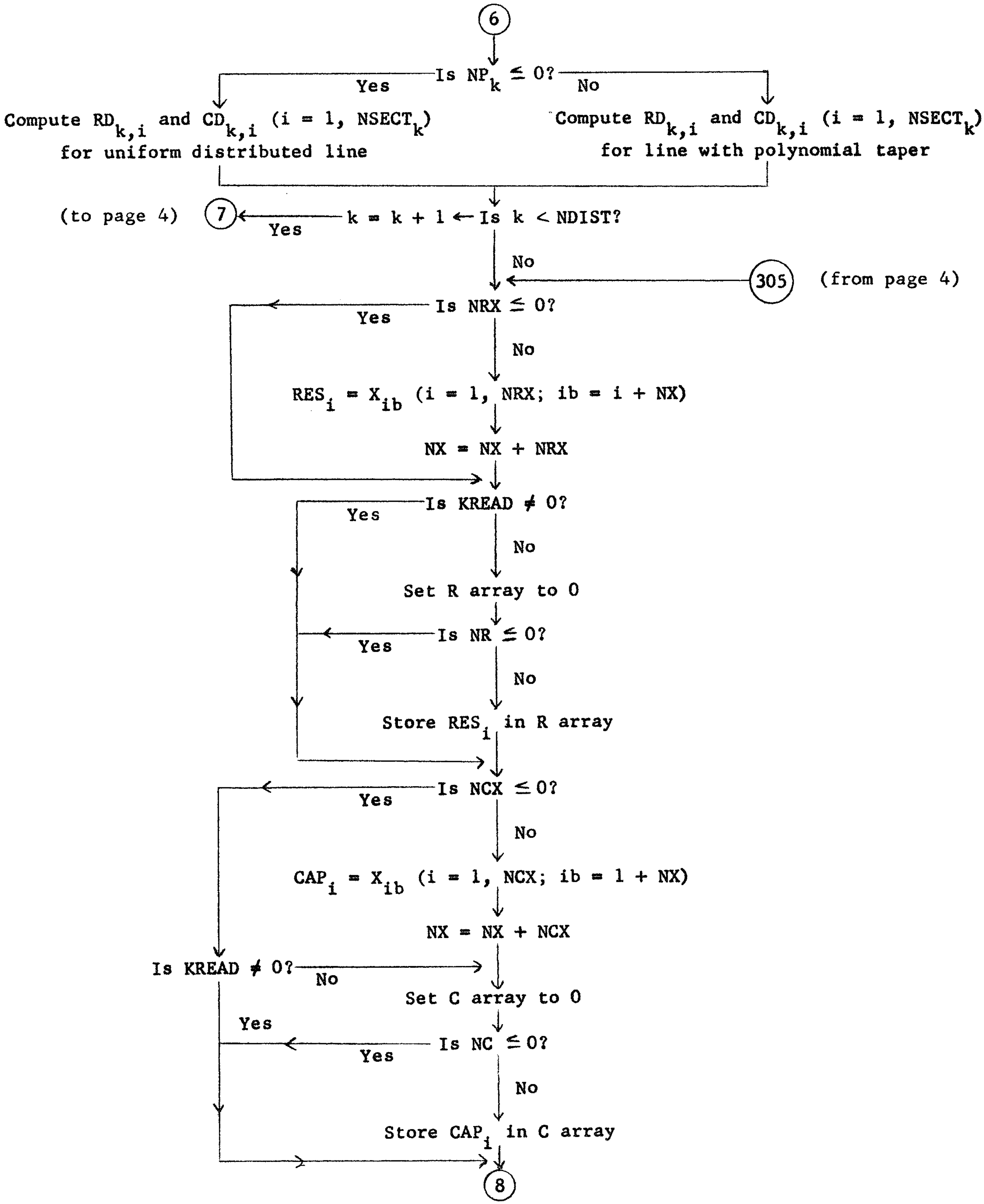
(from page 4)



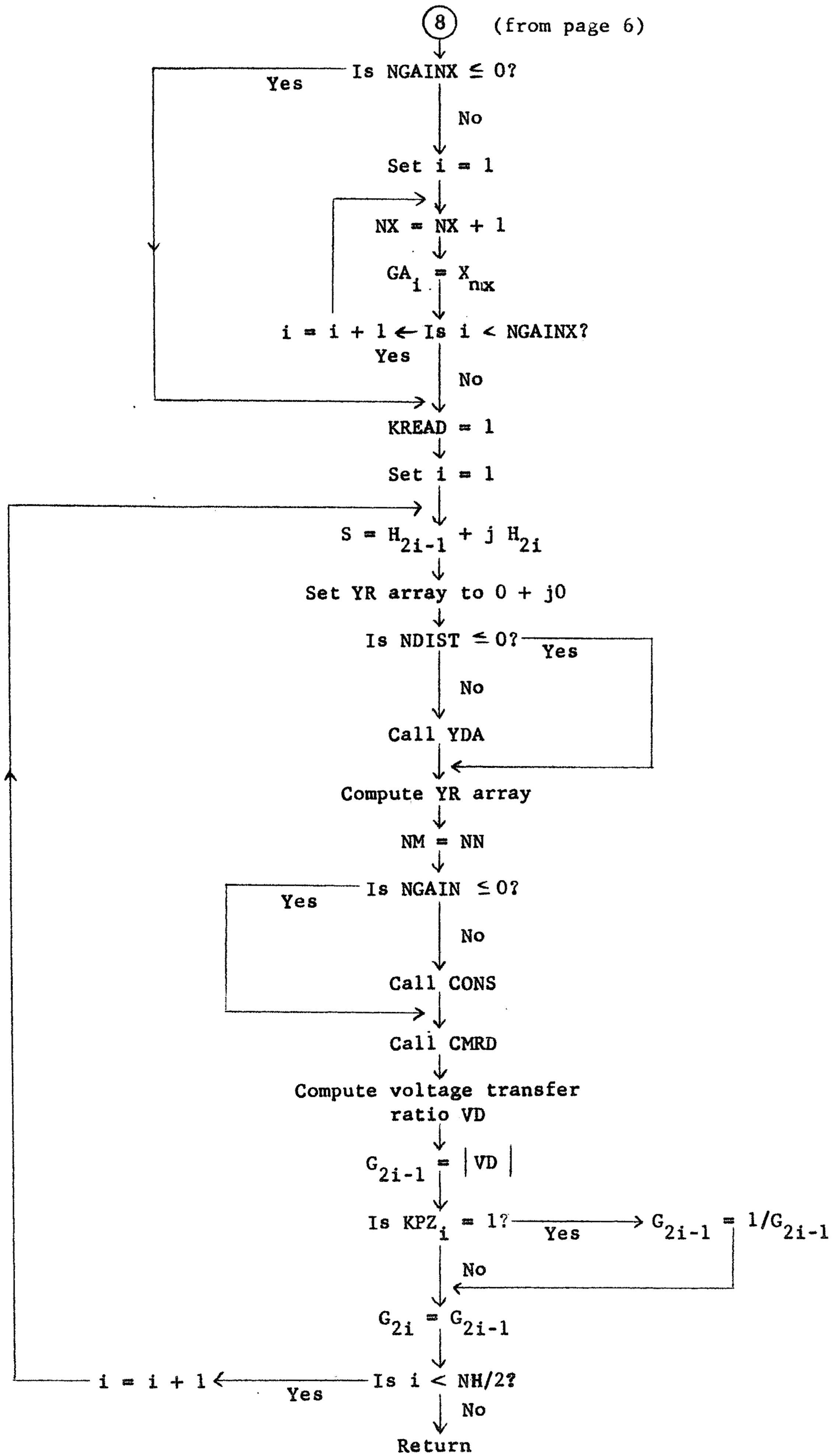
(to page 6)

Flow Chart Page 6

(from page 5)



(to page 7)


8
(from page 6)
Is NGAINX ≤ 0?
Yes
No
Set i = 1
NX = NX + 1
$GA_i = X_{nx}$
i = i + 1
Is i < NGAINX?
Yes
No
KREAD = 1
Set i = 1
$S = H_{2i-1} + j H_{2i}$
Set YR array to 0 + j0
Is NDIST ≤ 0?
Yes
No
Call YDA
Compute YR array
NM = NN
Is NGAIN ≤ 0?
Yes
No
Call CONS
Call CMRD
Compute voltage transfer ratio VD
$G_{2i-1} = |VD|$
Is KPZ_i = 1?
Yes
$G_{2i-1} = 1/G_{2i-1}$
No
$G_{2i} = G_{2i-1}$
i = i + 1
Is i < NH/2?
Yes
No
Return

```
              SUBROUTINE ANLYZ
        C     ANALYSIS SUBROUTINE FOR DLA NETWORK, 9-66
        C     REQUIRES SUBROUTINES YDA,CARD,CONS
000002        COMMON /OPT/X(20),XL(20),XU(20),KX(20),XP(20),H(20),G(20),W(20)
             1,G(20),PARAM(10,7),NOPT(10,10),N,NH,IERR,ITER,ERMIN,ITMAX,ALFA(8)
  0002        COMMON /DLA/YR(10,10),RO(5,50),CD(5,50),ND1(5),ND2(5),ND3(5),
             1NSECT(5),NM,GA(5),S,NG1(5),NG2(5),NG3(5),NG4(5),JL(5),
             2NN,NGAIN,NDIST
000002        COMPLEX YR,S
000002        DIMENSION NR1(10),NR2(10),NC1(10),NC2(10),RES(10),CAP(10),
             1R(10,10),C(10,10),P(5,5),KPZ(10),NP(5),NPX(5),NDRX(5),NDCX(5),
             2XT(5),RO(5),CO(5)
000002        COMPLEX VD
000002        DATA KREAD/0/
000002        IP=1
000003        IF (KREAD.EQ.1) GO TO 265
000005        NH2=NH/2
000007        READ 105, (KPZ(I),I=1,NH2)
000021        READ 105,NN,NDIST,NR,NC,NGAIN,NRX,NCX,NGAINX
000045    105 FORMAT (20I3)
000045        IF (NDIST.LE.0) GO TO 107
000047        DO 106 I=1,NDIST
000050        READ 105, ND1(I),ND2(I),ND3(I),NSECT(I),NP(I),NPX(I),
             1NDRX(I),NDCX(I)
000103    106 CONTINUE
000106    107 PRINT 405,NH2
000114    405 FORMAT (//1X*FIRST CALL OF DLA ANLYZ, OPTIMIZATION AT*I3
             1* FREQUENCIES*/)
000114        DO 404 I=1,NH2
000116        IF (KPZ(I).EQ.1) GO TO 402
  0121        PRINT 401,I,H(2*I-1),H(2*I)
000134    401 FORMAT (1X*FREQUENCY*I3* IS A ZERO LOCATED AT*E10.2,2H+J,E9.2)
000134        GO TO 404
000135    402 PRINT 403,I,H(2*I-1),H(2*I)
000151    403 FORMAT (1X*FREQUENCY*I3* IS A POLE LOCATED AT*E10.2,2H+J,E9.2)
000151    404 CONTINUE
000154        PRINT 406,NN
000161    406 FORMAT (
             11X*NUMBER OF NODES (REFERENCE NODE NOT INCLUDED)..............*I
000161        IF (NDIST.LE.0) GO TO 202
000163        DO 201 L=1,NDIST
000164        PRINT 125,L
000171    125 FORMAT (*0DISTRIBUTED NETWORK*I2)
000171        PRINT 127,NSECT(L),ND1(L),ND2(L),ND3(L),NDRX(L),NDCX(L),NP(L)
000222    127 FORMAT (
             21X*NUMBER OF SECTIONS IN DISTRIBUTED NETWORK..................*I
             31X*DISTRIBUTED NETWORK CONNECTED TO NODES...............*3I3
             41X*DISTRIBUTED R OPTIMIZED (1=YES, 0=NO)......................*I
             51X*DISTRIBUTED C OPTIMIZED (1=YES, 0=NO)......................*I
             61X*NUMBER OF TAPER COEFFICIENTS (0=UNIFORM LINE)..............*
000222        IF (ND1(L).LE.0) ND1(L)=NN+1
000230        IF (ND2(L).LE.0) ND2(L)=NN+1
000236        IF (ND3(L).LE.0) ND3(L)=NN+1
000244        IF (NP(L).LE.0) GO TO 140
000247        READ 155,XT(L)
000255        PRINT 407,NPX(L),XT(L)
  0267    407 FORMAT (
```

```
         11X*NUMBER OF TAPER COEFFICIENTS OPTIMIZED....................*I3
         21X*TOTAL LENGTH OF DISTRIBUTED LINE..................*E11.4
000267   140 IF (NPX(L).LE.0) GO TO 150
000272       I2=NPX(L)
000274       DO 142 I=1,I2
000275       IA=IP+I-1
  0277       PRINT 145,IA,I,X(IA)
000312   142 CONTINUE
000315       IP=IP+NPX(L)
000317   145 FORMAT (
         11X*X(*I2*) - OPTIMIZED TAPER COEFFICIENT*I3*.............*E11.4
000317   150 IF (NP(L).LE.NPX(L)) GO TO 165
000323       IPA=1+NPX(L)
000326       I2=NP(L)
000330       READ 155,(P(L,I),I=IPA,I2)
000344   155 FORMAT (8F10.0)
000344       PRINT 160,(I,P(L,I),I=IPA,I2)
000362   160 FORMAT (9X, *FIXED   TAPER   COEFFICIENT*I3*.............*E11.4
000362   165 IF (NDRX(L).LE.0) GO TO 175
000365       PRINT 170, IP,X(IP)
000375   170 FORMAT (
         11X*X(*I2*) - OPTIMIZED DISTRIBUTED RESISTANCE...........* E11.4
000375       IP=IP+1
000377       GO TO 185
000377   175 READ 155, RO(L)
000406       PRINT 180, RO(L)
000415   180 FORMAT (9X, *FIXED DISTRIBUTED RESISTANCE..............* E11.4
000415   185 IF (NDCX(L).LE.0) GO TO 195
000420       PRINT 190, IP,X(IP)
000430   190 FORMAT (
         11X*X(*I2*) - OPTIMIZED DISTRIBUTED CAPACITANCE..........* E11.4
  0430       IP=IP+1
000432       GO TO 201
000432   195 READ 155,CO(L)
000441       PRINT 200,CO(L)
000450   200 FORMAT (9X, *FIXED DISTRIBUTED CAPACITANCE.............* E11.4
000450   201 CONTINUE
000453   202 IF (NR.LE.0) GO TO 130
000455       READ 105,(NR1(I),NR2(I),I=1,NR)
000472   130 IF (NC.LE.0) GO TO 137
000474       READ 105, (NC1(I),NC2(I),I=1,NC)
000511   137 IF (NGAIN.LE.0) GO TO 205
000513       DO 139 I=1,NGAIN
000514       READ 105,NG1(I),NG2(I),NG3(I),NG4(I),JL(I)
000536   139 CONTINUE
000541   205 PRINT 138
000545   138 FORMAT ( )
000545       PRINT 410,NR,NRX,NC,NCX
000561   410 FORMAT (
          11X*NUMBER OF LUMPED RESISTORS...............................*I
          21X*NUMBER OF LUMPED RESISTORS OPTIMIZED.....................*I
          31X*NUMBER OF LUMPED CAPACITORS..............................*I
          41X*NUMBER OF LUMPED CAPACITORS OPTIMIZED....................*I
000561       IF (NRX.LE.0) GO TO 215
000563       DO 212 I=1,NRX
000564       IA=IP+I-1
000566       PRINT 211,IA,I,NR1(I),NR2(I),X(IA)
  0607   212 CONTINUE
```

```
000612      211 FORMAT (
               11X*X(*I2*) - OPTIMIZED LUMPED RESISTOR*I3* NODES*2I3*...*E11.4
000612          IP=IP+NRX
000613      215 IF (NR.LE.NRX) GO TO 225
000616          IA=NRX+1
000617          READ 155, (RES(I),I=IA,NR)
   0632         PRINT 220, (I,NR1(I),NR2(I),RES(I),I=IA,NR)
000655      220 FORMAT (9X  *FIXED   LUMPED   RESISTOR*I3* NODES*2I3*...*E11.4
000655      225 IF (NCX.LE.0) GO TO 235
000657          DO 232 I=1,NCX
000660          IA=IP+I-1
000662          PRINT 233,IA,I,NC1(I),NC2(I),X(IA)
000703      232 CONTINUE
000706      233 FORMAT (
               11X*X(*I2*) - OPTIMIZED LUMPED CAPACITOR*I3* NODES*2I3*..*E11.4
000706          IP=IP+NCX
000707      235 IF (NC.LE.NCX) GO TO 245
000712          IA=NCX+1
000713          READ 155, (CAP(I),I=IA,NC)
000726          PRINT 240, (I,NC1(I),NC2(I),CAP(I),I=IA,NC)
000751      240 FORMAT (9X  *FIXED   LUMPED   CAPACITOR*I3* NODES*2I3*..*E11.4
000751      245 PRINT 138
000755          IF (NGAINX.LE.0) GO TO 255
000757          DO 250 I=1, NGAINX
000760          IA=IP+I-1
000762          PRINT 251,IA,I,NG1(I),NG2(I),NG3(I),NG4(I),JL(I),X(IA)
001014      250 CONTINUE
001017      251 FORMAT (
               11X*X(*I2*) - OPTIMIZED VCVS*I3*, OUTPUT NODES*2I3/
               214X*INPUT NODES*2I3*, NODE*I3* ELIM.......*E11.4)
001017          IP=IP+NGAINX
  1020      255 IF (NGAIN.LE.NGAINX) GO TO 265
001023          IA=NGAINX+1
001024          READ 155, (GA(I),I=IA,NGAIN)
001037          PRINT 258,(I,NG1(I),NG2(I),NG3(I),NG4(I),JL(I),GA(I),I=IA,NGAIN)
001073      258 FORMAT (9X*FIXED VCVS*I3*, OUTPUT NODES*2I3/
               114X*INPUT NODES*2I3*, NODE*I3* ELIM.......*E11.4)
001073      265 NX=0
001074          IF(NDIST.LE.0) GO TO 305
001076          DO 299 L=1,NDIST
001077          KD=0
001100          IF (NPX(L).LE.0) GO TO 275
001102          I2=NPX(L)
001104          DO 270 I=1,I2
001106          NX=NX+1
001110          P(L,I)=X(NX)
001114      270 CONTINUE
001116          KD=1
001117      275 IF (NDRX(L).LE.0) GO TO 280
001122          NX=NX+1
001123          RO(L)=X(NX)
001126          KD=1
001127      280 IF (NDCX(L).LE.0) GO TO 285
001132          NX=NX+1
001133          CO(L)=X(NX)
001136          KD=1
001137      285 IF (KREAD.EQ.0) GO TO 290
  1140          IF (KD.EQ.0) GO TO 299
```

```
001141      290 DV=NSECT(L)
001144          ROD=RO(L)/DV
001146          COD=CO(L)/DV
001150          IF (NP(L).LE.0) GO TO 296
001152          DM=NSECT(L)-1
001155          I2=NSECT(L)
  1157          DO 294 I=1,I2
001160          DI=I-1
001162          XD=DI*AT(L)/DM
001166          AP=1.
001167          I3=NP(L)
001171          DO 292 J=1,I3
001173          AP=AP+P(L,J)*XD**J
001205      292 CONTINUE
001210          RD(L,I)=ROD*AP
001214      294 CD(L,I)=COD/AP
001222          GO TO 299
001222      296 I2=NSECT(L)
001225          DO 298 I=1,I2
001226          RD(L,I)=ROD
001232      298 CD(L,I)=COD
001240      299 CONTINUE
001243      305 IF (NRX.LE.0) GO TO 315
001245          DO 310 I=1,NRX
001246          NX=NX+1
001250      310 RES(I)=X(NX)
001255          GO TO 320
001255      315 IF (KREAD.GT.0) GO TO 340
001260      320 DO 325 I=1,NN
001262          DO 325 J=1,NN
001263          R(I,J)=0.
  1267      325 CONTINUE
001274          IF (NR.LE.0) GO TO 340
001275          DO 335 I=1,NR
001277          COND=1./RES(I)
001302          KR1=NR1(I)
001304          KR2=NR2(I)
001306          IF (KR2.LE.0) GO TO 330
001307          R(KR2,KR2)=COND+R(KR2,KR2)
001313          R(KR2,KR1)=-COND+R(KR2,KR1)
001320          R(KR1,KR2)=R(KR2,KR1)
001325      330 R(KR1,KR1)=COND+R(KR1,KR1)
001332      335 CONTINUE
001334      340 IF (NCX.LE.0) GO TO 350
001336          DO 345 I=1,NCX
001337          NX=NX+1
001341      345 CAP(I)=X(NX)
001346          GO TO 355
001346      350 IF (KREAD.GT.0) GO TO 375
001351      355 DO 360 I=1,NN
001353          DO 360 J=1,NN
001354          C(I,J)=0.
001360      360 CONTINUE
001365          IF (NC.LE.0) GO TO 375
001366          DO 370 I=1,NC
001370          KC1=NC1(I)
001372          KC2=NC2(I)
  1374          IF (KC2.LE.0) GO TO 365
```

```
001375          C(KC2,KC2)=CAP(I)+C(KC2,KC2)
001402          C(KC2,KC1)=-CAP(I)+C(KC2,KC1)
001407          C(KC1,KC2)=C(KC2,KC1)
001415      365 C(KC1,KC1)=CAP(I)+C(KC1,KC1)
001423      370 CONTINUE
001425      375 IF (NGAINX.LE.0) GO TO 380
001427          DO 377 I=1,NGAINX
001430          NX=NX+1
001432      377 GA(I)=X(NX)
001437      380 KREAD=1
001440          DO 400 L=1,NH2
001442          L2=2*L
001443          L2M=L2-1
001445          S=CMPLX(H(L2M),H(L2))
001452          DO 385 I=1,NN
001453          DO 385 J=1,NN
001454          YR(I,J)=(0.,0.)
001463      385 CONTINUE
001470          IF (NDIST.LE.0) GO TO 387
001472          CALL YDA
001473      387 DO 390 I=1,NN
001475          DO 390 J=1,NN
001476          YR(I,J)=YR(I,J)+R(I,J)+C(I,J)*S
001521      390 CONTINUE
001526          NM=NN
001527          IF (NGAIN.LE.0) GO TO 395
001530          CALL CONS
001531      395 CALL CMRO
001532          VD=-YR(2,1)/YR(2,2)
001542          G(L2M)=CABS(VD)
001546          IF (KPZ(L).EQ.1) G(L2M)=1./G(L2M)
001553      400 G(L2)=G(L2M)
001561          RETURN
001562          END
```

```
                SUBROUTINE CMRD
000002          COMMON /DLA/YR(10,10),RD(5,50),CD(5,50),ND1(5),ND2(5),ND3(5),
               1NSECT(5),NM,GA(5),S,NG1(5),NG2(5),NG3(5),NG4(5),JL(5),
               2NN,NGAIN,NOIST
000002          COMPLEX YR,S
  0002          NA=NN-NGAIN
000004          NZ=NN-(2+NGAIN)
000006          IF (NZ.LT.1) RETURN
000011          DO 110 K=1,NZ
000013          NP=NA-1
000015          DO 105 J=1,NP
000017          DO 105 I=1,NP
000020          YR(I,J)=YR(I,J)-YR(I,NA)*YR(NA,J)/YR(NA,NA)
000052      105 CONTINUE
000057      110 NA=NA-1
000063          RETURN
000063          END
```

```
              SUBROUTINE YOA
000002        COMMON /DLA/YR(10,10),RD(5,50),CD(5,50),ND1(5),ND2(5),ND3(5),
             1NSECT(5),NM,GA(5),S,NG1(5),NG2(5),NG3(5),NG4(5),JL(5),
             2NN,NGAIN,NDIST
000002        COMPLEX YR,S
000002        COMPLEX AR(2,2),BR(2,2),CR(2,2),DR,ER,GR
000002        DO 125 LZ=1,NDIST
000004    109 DO 112 I=1,2
000006        DO 110 J=1,2
000007        CR(I,J)=(0.,0.)
000015    110 CONTINUE
000020        CR(I,I)=(1.,0.)
000025    112 CONTINUE
000027        I2=NSECT(LZ)
000031        DO 120 I=1,I2
000033        DO 115 IA=1,2
000034        DO 115 IB=1,2
000035        AR(IA,IB)=CR(IA,IB)
000046    115 CONTINUE
000053        BR(1,1)=(1.,0.)+CD(LZ,I)*RD(LZ,I)*S
000070        BR(1,2)=CMPLX (RD(LZ,I),0.)
000075        BR(2,1)=CD(LZ,I)*S
000105        BR(2,2)=(1.,0.)
000107        CR(1,1)=AR(1,1)*BR(1,1)+AR(1,2)*BR(2,1)
000123        CR(1,2)=AR(1,1)*BR(1,2)+AR(1,2)*BR(2,2)
000136        CR(2,1)=AR(2,1)*BR(1,1)+AR(2,2)*BR(2,1)
000152        CR(2,2)=AR(2,1)*BR(1,2)+AR(2,2)*BR(2,2)
000165    120 CONTINUE
000170        DR=(1.,0.)/CR(1,2)
000177        ER=DR*CR(2,2)
000205        GR=DR*CR(1,1)
000213    121 L=ND1(LZ)
000216        M=ND2(LZ)
000220        N=ND3(LZ)
000222        YR(L,L)=ER+YR(L,L)
000232        YR(M,M)=GR+YR(M,M)
000242        YR(L,M)=-DR+YR(L,M)
000252        YR(M,L)=YR(L,M)
000262        YR(N,L)=DR-ER+YR(N,L)
000274        YR(L,N)=YR(N,L)
000305        YR(N,M)=DR-GR+YR(N,M)
000317        YR(M,N)=YR(N,M)
000330        YR(N,N)=ER+GR-(2.,0.)*DR+YR(N,N)
000347    125 CONTINUE
000351        RETURN
000352        END
```

```
              SUBROUTINE CONS
         C    LPH SUBROUTINE 235 REPLACES PERRY 233, JULY 1968
000002        COMMON /LLA/YR(10,10),RD(5,50),CD(5,50),ND1(5),ND2(5),ND3(5),
             1NSECT(5),NM,GA(5),S,NG1(5),NG2(5),NG3(5),NG4(5),JL(5),
             2NN,NGAIN,NDIST
000002        COMPLEX YR,S
000002        DATA NREP/0/
000002        DIMENSION NRE(10),NCE(10)
000002        NM=NN-NGAIN
000004        IF (NREP.GT.0) GO TO 121
000007        DO 110 I=1,NN
000010        NRE(I)=0
000012    110 NCE(I)=0
000015        DO 120 J=1,NGAIN
000017        DO 120 I=1,NN
000020        IF (I.EQ.NG1(J)) NRE(I)=1
000024        IF (I.EQ.JL(J)) NCE(I)=1
000031    120 CONTINUE
000036        NREP=1
000037    121 IN=1
000040        DO 145 I=1,NM
000042    123 IF (NRE(IN)) 130,130,125
000045    125 IN=IN+1
000047        GO TO 123
000047    130 IF (IN.EQ.I) GO TO 140
000051    132 DO 135 J=1,NN
000053    135 YR(I,J)=YR(IN,J)
000070    140 IN=IN+1
000072    145 CONTINUE
000074        DO 165 I=1,NGAIN
000076        K=NG1(I)
000100        N=NG4(I)
000102        L=NG3(I)
000104        M=JL(I)
000106        IF (N.EQ.0) GO TO 148
000107        IF (M.EQ.K) GO TO 151
000111        IF (M.EQ.L) GO TO 153
000112        DO 146 J= 1,NM
000114    146 YR(J,L)=YR(J,L)+YR(J,N)
000133        DO 147 J=1,NM
000135    147 YR(J,K)=YR(J,K)-YR(J,N)/GA(I)
000160        GO TO 165
000161    151 DO 152 J=1,NM
000163    152 YR(J,N)=YR(J,N)-YR(J,K)*GA(I)
000205        GO TO 149
000206    153 DO 154 J=1,NM
000210    154 YR(J,N)=YR(J,N)+YR(J,L)
000227        GO TO 155
000230    148 IF (L.EQ.M) GO TO 155
000232    149 DO 150 J=1,NM
000234    150 YR(J,L)=YR(J,L)+YR(J,K)*GA(I)
000256        GO TO 165
000257    155 DO 160 J=1,NM
000261    160 YR(J,K)=YR(J,K)+YR(J,L)/GA(I)
000304    165 CONTINUE
000307        IN=1
000310        DO 190 I=1,NM
```

```
000311      172 IF (NCE(IN)) 175,175,170
000314      170 IN=IN+1
000316          GO TO 172
000316      175 IF (IN.EQ.I) GO TO 185
000320          DO 180 J=1,NM
000322      180 YR(J,I)=YR(J,IN)
000337      185 IN=IN+1
000341      190 CONTINUE
000343          RETURN
000344          END
```

Input Data Format for Subroutine ANLYZ for DLA Network

Card No.	Variable	Columns	Input Format	Purpose
1	KPZ(I)	1-30	10I3	Indicators for whether the ith frequency is a zero (KPX(I) = 0) or a pole (KPZ(I) = 1). (i=1,10)
2	NN	1-3	I3	Number of nodes (not counting reference node) in DLA network (0-9)
	NDIST	4-6	I3	Number of distributed networks (0-5)
	NR	7-9	I3	Number of lumped resistors in DLA network (0-10)
	NC	10-12	I3	Number of lumped capacitors in DLA network (0-10)
	NGAIN	13-15	I3	Number of VCVS gain elements (0-5)
	NRX	16-18	I3	Number of lumped resistors optimized (0-5)
	NCX	19-21	I3	Number of lumped capacitors optimized (0-5)
	NGAINX	22-24	I3	Number of VCVS gain elements optimized (0-5)

The following card is required NDIST times

Card No.	Variable	Columns	Input Format	Purpose
3	ND1(I)	1-3	I3	Node number for terminal 1 of Ith distributed network
	ND2(I)	4-6	I3	Node number for terminal 2 of Ith distributed network
	ND3(I)	7-9	I3	Node number for terminal 3 of Ith distributed network
	NSECT(I)	10-12	I3	Number of sections used in lumped model for Ith distributed network
	NP(I)	13-15	I3	Number of polynomial coefficients used in describing taper of Ith distributed network (0=uniform line) (0-5)

	NPX(I)	16-18	I3	Number of polynomial coefficients optimized in Ith distributed line (0-5)
	NDRX(I)	19-21	I3	Indicator for optimizing resistance of Ith distributed line (1 = optimize)
	NDCX(I)	22-24	I3	Indicator for optimizing capacitance of Ith distributed line (1 = optimize)

The following card group consisting of cards 4-7 is required NDIST times for values of I from 1 to NDIST.

The following card group consisting of cards 4-5 is required only if NP(I) > 0

4	XT(I)	1-10	E10.0	Total length of Ith distributed line

The following card required only if NP(I) > NPX(I)

5	P(I,J)	1-80	8E10.0	Values of non-optimized coefficients for Ith distributed line (J=1+NPX(I), NP(I))

The following card required only if NDRX(I) = 0

6	RO(I)	1-10	E10.0	Value of resistance for the distributed network (total resistance if line is uniform)

The following card required only if NDCX(I) = 0

7	CO(I)	1-10	E10.0	Value of capacitance for Ith distributed network (total capacitance if line is uniform)

The following card required only if NR > 0

8	NR1(I), NR2(I)	1-60	20I3	Node numbers to which the Ith resistor is connected (I=1, NR)

The following card required only if NC > 0

9	NC1(I), NC2(I)	1-60	20I3	Numbers of nodes to which the Ith capacitor is connected (I=1, NC)

The following card is required NGAIN times

10	NG1(I)	1-3	I3	Number of node to which source of Ith VCVS is connected (other end of source is assumed connected to ground)
	NG3(I)	7-9	I3	Number of node to which positive (noninverting) input terminal for Ith VCVS is located
	NG4(I)	10-12	I3	Number of node to which negative (inverting) input terminal for Ith VCVS is located (this should be o if only a single VCVS input is used)
	JL(I)	13-15	I3	Number of node which it is desired to eliminate [either NG1(I), NG3(I), or NG4(I)]

The following card required only if NR > NRX

11	RES(I)	1-80	8E10.0	Values of lumped fixed resistors R_i (I = NRX + 1, NR)

The following card required only if NC > NCX

12	CAP(I)	1-80	8E10.0	Values of lumped fixed capacitors C_i (I = NCX + 1, NC)

The following card required only if NGAIN > NGAINX

13	GA(I)	1-80	8E10.0	Value of unoptimized VCVS gain for Ith source (I = NGAINX + 1, NGAIN)

Order in which variables X(I) are assigned to elements of DLA network

Order	Condition	Variables Assigned	Number of Variables
Assignments number 1-3 are made NDIST times for values of I from 1 to NDIST			
1	NPX(I) > 0	P(I,J)	J = 1, NPX(I)
2	NDRX(I) = 1	RO(I)	1
3	NDCX(I) = 1	CO(I)	1
4	NRX > 0	RES(I)	I = 1, NRX
5	NCX > 0	CAP(I)	I = 1, NCX
6	NGAINX > 0	GA(I)	I = 1, NGAINX