N74-14601

NASTRAN MULTIPARTITIONING AND "ONE-SHOT" SUBSTRUCTURING

By Alvin Levy

Grumman Aerospace Corporation, Bethpage, New York

SUMMARY

For intermediate size problems where all the data is acces-
ıle, the present method of substructuring in three separate phases
or static analysis) is unneccessarily cumbersome. The versa-
.ity of NASTRAN's DMAP and internal logic lends itself to find-
; a practical alternative to these procedures whereby self-
ıtained special-purpose ALTER packages can be written to be run
one pass. Two examples are presented here under the titles of
.tipartitioning and "one-shot" substructuring. The flow of
.tipartitioning resembles that of the present three-phase sub-
ructuring. The basic effect is to partition the structure
to substructures and operate on each substructure separately.
is can be used to reduce the bandwidth of a given problem as
ll as to store information which will allow a change to be made
one of the substructures in a later run. This latter pro-
dure is carried out in a second program titled "one-shot" sub-
ructuring.

INTRODUCTION

At present, in order to use NASTRAN substructuring for a
atic analysis, the user must perform a three-phase analysis on
e structure as discussed in reference 1. In Phase I, the stiff-
ss and load matrices are computed and saved for each substruc-
re. This requires a separate computer run for each substruc-
re. Phase II merges the reduced matrices from Phase I and
mputes the substructure boundary (a-set) displacements. This
quires the input of one tape for each substructure, an ALTER
ckage to suit the given problem, and user-generated partition
ctors or multipoint constraints. In Phase III, each substruc-
re is restarted by using as input the a-set displacements computed
Phase II and Phase III gives as output the final solution. Once
ain, this procedure requires a separate run for each substructure.

One of the useful applications of substructuring is to allow the user to make changes to one or more of the substructures and regenerate solutions with a minimum of man and machine effort. This application requires the user to execute one Phase I run for each substructure change, one Phase II run, and as many Phase III runs as there are total substructures.

Some of the practical difficulties encountered at present are summarized as follows:

1) Each phase must be run consecutively and this increases the real-time requirements.

2) For Phases I and III each substructure must be run independently. This increases the cost.

3) The user must take care in handling the tapes and restart dictionaries used in the various phases.

4) For Phase II, the user must write a DMAP ALTER to suit the given problem. This requires taking into account the number of substructures involved. The user must also input a partition vector for each substructure.

5) If a substructure is changed and the problem rerun, the three phases must be run consecutively once again.

For many large-scale problems encountered, especially where information is gathered at different locations, this procedure will be practical, but for many cases of intermediate and large size problems where all the data is accessible and fits within the storage capacity of the computer, this procedure seems unnecessarily cumbersome.

A practical alternative to these procedures is to write special purpose programs through the use of DMAP ALTER packages, each suitable for a given need and each self contained in one program to be run in one pass. Examples of this are presented in the present paper and are titled multipartitioning and "one-shot" substructuring. These procedures contain the following features:

1) Only DMAP ALTER statements are involved so that no additional capabilities need be included in NASTRAN, although some are suggested in order to make the methods more effi ient and flexible.

2) The complete substructuring (multipartitioning) analysis can be carried out in one run.

3) If a change is made in one of the substructures the program only requires as input the changed substructure and again gives a complete analysis of the entire structure in one run.

4) The rules for setting up the program for a given problem do not require the user to make any changes in the DMAP ALTER package.

5) The need for partition vectors has been eliminated.

6) The instructions to be followed for generating the required Case Control and Bulk Data Decks are simple.

7) There is a minimum of tape handling and no restart dictionaries required.

At present the ALTER packages presented here for static analysis carry the limitation that when making changes to a given substructure for a follow-up substructure analysis, the elements adjacent to the boundary points (a-set) must be unchanged in stiffness because the contribution due to these elements cannot be distinguished in forming $[\bar{K}_{aa}]$. (See fig. 1.)

MULTIPARTITIONING AND ONE-SHOT SUBSTRUCTURING

The program flows for multipartitioning and one-shot substructuring are given in figure 1 and the specific DMAP ALTER packages are given in figures 2 and 3. In the multipartitioning package the entire data for a complete structure is given as input, along with the a-set points used to partition the structure and the grid points which are contained in each substructure. The program then partitions the stiffness matrix by isolating each partitioned substructure. The individual substructures are then operated on separately as if the boundary degrees of freedom (a-set degrees of freedom) are completely fixed. The interaction effects are then summed and the a-set points are solved for. This information is then passed back to each substructure and the solution for each substructure is carried out. It can be seen that this method follows the normal procedure of substructuring without having to form partition vectors for each substructure. The overall effect is to partition the stiffness matrix as would normally be done by using the partitioning (ASET or OMIT) feature, but the name multipartitioning comes from the similarity between the present method and the method of partitioning coupled with resequencing of nodes which results in a reduction of the bandwidth. A demonstration of this method is shown in figure 4. Figure 4(a) represents a finite-element idealization where the nodes are numbered to produce the minimum bandwidth. The idealization is partitioned into two sections as shown. Figure 4(b) represents the initial structure before partitioning and figure 4(c) after partitioning. The bandwidth has been reduced from 7 (assuming one degree of freedom per node) to 6. If we use the methods of multipartitioning the two subdivisions are treated separately; thus, the bandwidth is reduced to 4 as shown in figure 4(d). This reduction could also be accomplished by using the method of partitioning (fig. 4(c)) along with reordering the nodes as shown in figure 4(d).

We can now make a change in one of the substructures and repeat the analysis. Only the data for the changed substructure is required, along with the stored information (on tape) of the old complete structure. Required calculations for the new structure are carried out (see fig. 1) and the complete analysis of all the substructures proceeds as before.

USER PROCEDURES

The user procedures will be given by demonstration. Figure 5(a) represents a structure to be analyzed by the present methods. The structure is subdivided into three substructures and we are interested in two different loading conditions. The three substructures are shown in figure 5(b). (An intermediate solution for the three substructures fixed at the a-set points is included in the ALTER package.)

The Executive Control Deck contains the multipartitioning ALTER package (fig. 2). The Case Control Deck and Bulk Data Deck are shown in figure 6. The Bulk Data Deck will be discussed first. The elements, grid points, loads, and property cards are as usual. The ASET card defines the boundary points of the substructures. The boundary conditions (simply supported in this case) are placed on the grid point identifications (for simplicity). All the grid points not included in the a-set are placed on SPC cards as follows: those contained in substructure i are placed in SPC set number 100 + i (see SPC and SPC1 cards) and then added together so that SPC set j contains all points not in substructure j (see SPCADD cards). This method is used in the program to partition out each substructure. One auxiliary device must be mounted (INPT) for the multipartitioning program. If information is to be retained for subsequent use (e.g., to change one of the substructures) then three additional tapes must be mounted (INP3, INP4, and INP5), and a parameter TAPE=1 must be defined. (See PARAM card.) With a slight modification to the DMAP ALTER packages, this can be reduced to one additional tape. The Case Control Deck first defines sets corresponding to the nodes and elements contained within each substructure. If these sets are omitted, the output for a given substructure will contain null values for quantities corresponding to nodes and elements not contained in the given substructure but contained in the total substructure. The subcase definitions are as follows: the first digit refers to the substructure and the second digit corresponds to the load case (e.g., SUBCASE 31 corresponds to substructure 3 load case 1). The subcase sequence is as shown, i.e., each subcase is defined as many times as there are loading conditions, where the number of subcases must be the same for each substructure. The subcase numbering system is only suggested as a mnemonic to be used for ordering the subcases correctl

Figure 5(c) represents a change in geometry of substructure 2. Any change can be made (geometry, material properties) so long as the elements adjacent to the a-set points remain unchanged in stiffness. The Executive Control Deck now contains the one-shot substructuring ALTER package (fig. 3). The Bulk Data Deck contains only the new substructure and load conditions along with the a-set points for the complete structure (fig. 7). The ASET card must contain the same number of degrees of freedom as in the original multipartitioning run and the a-set points contained in the changed substructure must occupy the same relative position as before. For this purpose fictitious grid points (or scalar points) must be defined and constrained on SPC cards. Two parameters are defined in the Bulk Data Deck. NUMSUB is set equal to the total number of substructures and SUBNUM is set equal to the number of the substructure to be changed. The Case Control Deck contains one subcase for each load condition and the number of load conditions must be the same as in the original multipartitioning run.

CONCLUDING REMARKS

It has been shown that using NASTRAN's DMAP capabilities one can write ALTER packages to handle special cases of substructuring to be run in a single pass, without the use of new modules. These programs result in a saving of computer cost and real time as well as lessening the chance of error due to data handling. However, greater versatility could be obtained if some additional capabilities were included in the NASTRAN program. These capabilities include set definitions for elements, grid points, and a-set (and o-set) degrees of freedom.

In connection with "partitioning" methods a recent publication (ref. 2) should be of interest.

REFERENCES

1. MacNeal, Richard H., ed.: The NASTRAN Theoretical Manual. NASA SP-221 (01), 1972.

2. Meyer, Christian: Solution of Linear Equations – State-of-the-Art. J. Struct. Div., ASCE, vol. 99, no. ST7, July 1973, pp. 1507-1526.

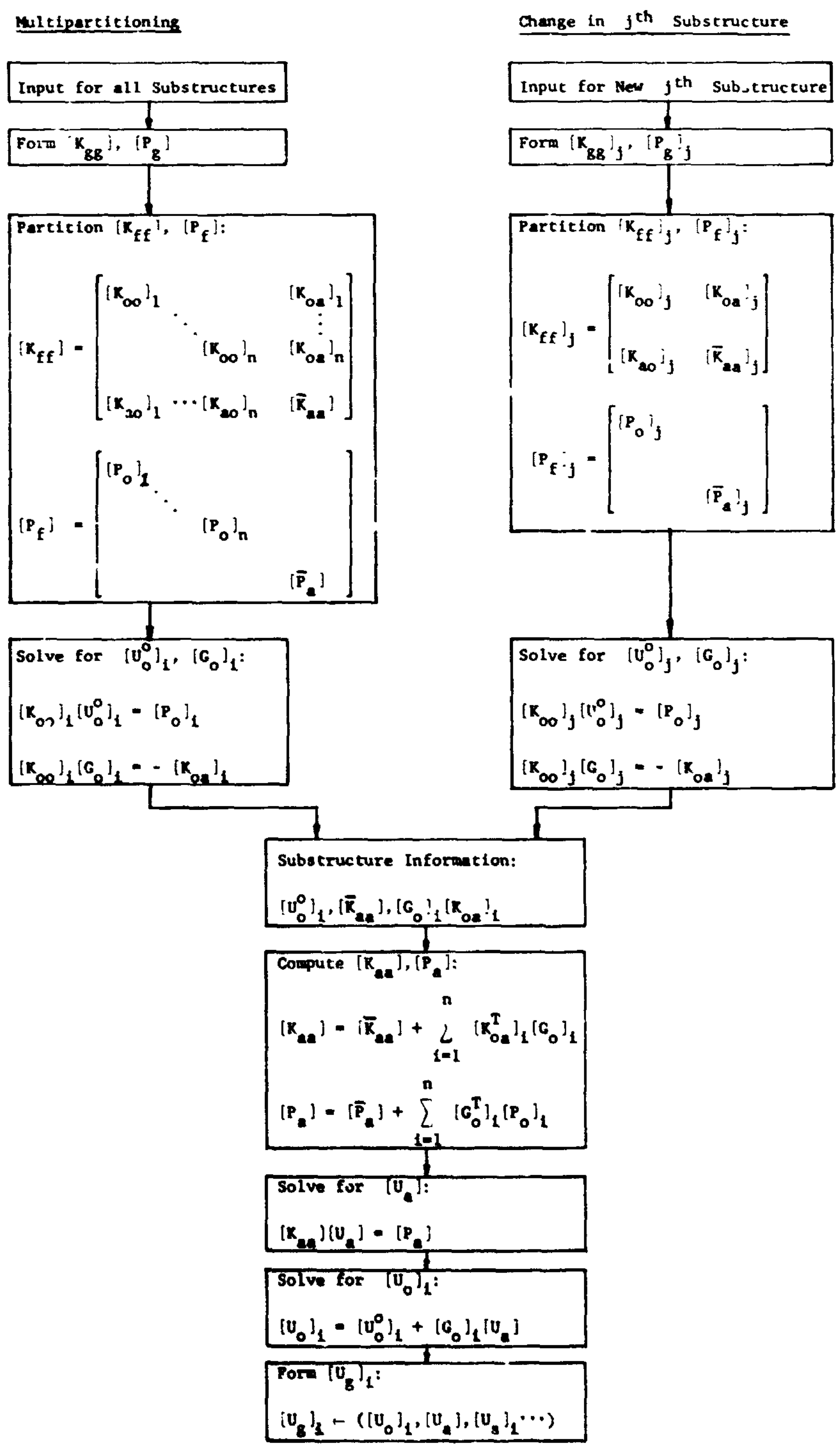


Figure 1.- Scheme for multipartitioning and one-shot substructuring.

```
ALTER 1,1
BEGIN NO.1 STATIC ANALYSIS-SERIES M1-MULTIPARTITIONING $
ALTER 50
PARAM //C,N,NOP/V,N,P1=-1
PARAM //C,N,NOP/V,N,TRUE=-1 $
PARAM //C,N,NOP/V,Y,TAPE=-1 $
$ INITIALIZE TAPE - WRITE LABEL AND REWIND
OUTPUT1, ,,,,//C,N,-1/C,N,0/C,N,TPO $
COND TPNO1,TAPE
$ INFORMATION TO BE SAVED FOR SUBSEQUENT RUNS- IF NOT DESIRED SET
$                      TAPE=-1 ON PARAM BULK DATA CARD
OUTPUT1, ,,,,//C,N,-1/C,N,3/C,N,TP3 $
OUTPUT1, ,,,,//C,N,-1/C,N,4/C,N,TP4 $
OUTPUT1, ,,,,//C,N,-1/C,N,5/C,N,TP5 $
LABEL TPNO1
ALTER 54
$  FORM PARTITION VECTOR G(L,COMP)
VEC USET/V/C,N,G/C,N,L/C,N,COMP $
CHKPNT V $
$  FORM PARTITION VECTOR F(L,COMP)
VEC USET/VFLC/C,N,F/C,N,L/C,N,COMP $
CHKPNT VFLC $
PRTPARM //C,N,0 $
ALTER 75
$  PARTITION OUT L-SET
$  L-SET IS SAME FOR ALL SUBSTRUCTURES
UPARTN USET,KFF/KLLB,,KAO,/C,N,F/C,N,A/C,N,0 $
CHKPNT KLLB,KAO $
ALTER 84
JUMP LBL7 $
ALTER 94
PARAM //C,N,SUB/V,N,NULL/C,N,-1/V,N,P1 $
COND LBLN1,NULL
$  INITIALIZE KLLT TO NULL , FIRST PASS ONLY
ADD KLL,/KLLT/C,N,(0.0,0.0) $
CHKPNT KLLT $
COND TPNO2,TAPE
OUTPUT1 KLLB,,,,//C,N,0/C,N,3 $
LABEL TPNO2
LABEL LBLN1
COND TPNO3,TAPE
OUTPUT1 KAO,GO,,,// C,N,0/C,N,3 $
LABEL TPNO3
ADD KLL,KLLT/KLLX/ $
CHKPNT KLLX $
EQUIV KLLX,KLLT/TRUE $
CHKPNT KLLT $
COND LBL5A,P1 $
$  SUBTRACT KLLB FROM KLLT EACH PASS AFTER THE FIRST
$  FIRST PASS GIVES TOTAL CONTRIBUTION OF KLLB
$  SINCE L-SET IS SAME FOR ALL SUBSTRUCTURES
ADD KLLT,KLLB/KLLXX/C,N,(1.0,0.0)/C,N,(-1.0,0.0) $
CHKPNT KLLXX $
EQUIV KLLXX,KLLT/TRUE $
CHKPNT KLLT $
LABEL LBL5A
ALTER 96
$  PARTITION OUT L-SET
$  L-SET IS SAME FOR ALL SUBSTRUCTURES
PARTN PG,,V/PLB,,,/C,N,1/C,N,2/C,N,2 $
CHKPNT PLB $
ALTER 100,101
SSG2 USET,GM,YS,KFS,GO,,PG/,PO,PS,PL $
CHKPNT PO,PS,PL $
```

Figure 2.- DMAP ALTER package for multipartitioning.

```
ALTER 102
COND LBLN2,NULL
$  INITIALIZE PLT TO NULL, FIRST PASS ONLY
ADD PL,/PLT/C,N,(0.0,0.0) $
CHKPNT PLT $
COND TPNO4,TAPE
OUTPUT1 PLB,,,,//C,N,0/C,N,4 $
LABEL TPNO4
LABEL LBLN2
TRNSP GO/GOT $
COND TPNO5,TAPE
OUTPUT1 GOT,PO,,,//C,N,0/C,N,4 $
LABEL TPNO5
ADD PL,PLT/PLX/ $
CHKPNT PLX $
EQUIV PLX,PLT/TRUE $
CHKPNT PLT $
COND LBL10A,P1 $
$  SUBTRACT PLB FROM PLT EACH PASS AFTER THE FIRST
$  FIRST PASS GIVES TOTAL CONTRIBUTION OF PLB
$  SINCE L-SET IS SAME FOR ALL SUBSTRUCTURES
ADD PLT,PLB/PLXX/C,N,(1.0,0.0)/C,N,(-1.0,0.0) $
CHKPNT PLXX $
EQUIV PLXX,PLT/TRUE $
CHKPNT PLT $
LABEL LBL10A
FBS LOO,UOO,PO/UOOV $
CHKPNT UOOV $
MATPRN UOOV,PO,,,//$
MATGPR GPL,USET,SIL,UOOV//C,N,O $
MATGPR GPL,USET,SIL,PO//C,N,O $
ALTER 103,111
PARAM //C,N,ADD/V,N,P1/V,N,P1/C,N,1 $
OUTPUT1 USET,UOOV,GO,PG,//C,N,0/C,N,0 $
ALTER 118
SOLVE KLLT,PLT/ULLB/C,N,1 $
CHKPNT ULLB $
MATPRN ULLB,PLT,KLLT,, // $
PARAM //C,N,NOP/V,N,NSKIP1=1 $
INPUTT1 /,,,,/C,N,-3/C,N,0/C,N,SUB
LABEL LBL110
INPUTT1 /USET1,UOOV1,GO4,PC1,/C,N,0/C,N,0 $
SDR1 USET1,PG1,ULLB,UOOV1,,GO4,,,,,/UGV,PGG,/V,N,NSKIP1/C,N,STATICS $
CHKPNT UGV,PGG $
MATPRN UGV,PGG,,,// $
PARAM //C,N,ADD/V,N,NSKIP1/V,N,NSKIP1/C,N,1 $
PARAM //C,N,SUB/V,N,P1/V,N,P1/C,N,1 $
COND LBL80,P1 $
REPT LBL110,100 $
JUMP ERROR1 $
LABEL LBL80 $
PARAM //C,N,NOT/V,N,TEST1/V,N,P1 $
COND ERROR5,TEST1 $
SDR2 CASECC,CSTM,MPT,,EQEXIN,SIL,,EDT,BGPDT,PGG,,UGV,EST,/OPG1,,OUGV1,
          OES1,OEF1,/C,N,STATICS $
OFP OPG1,OUGV1,OES1,OEF1,,//V,N,CARDNO $
SAVE CARDNO $
PRTPARM //C,N,0 $
COND TPNO9,TAPE
OUTPUT1 CASECC,CSTM,MPT,EQEXIN,SIL//C,N,0/C,N,5 $
OUTPUT1 EDT,BGPDT,PGG,EST,//C,N,0/C,N,5 $
INPUTT1 /,,,,/C,N,-3/C,N,3/C,N,TP3 $
INPUTT1 /,,,,/C,N,-3/C,N,4/C,N,TP4 $
INPUTT1 /,,,,/C,N,-3/C,N,5/C,N,TP5 $
LABEL TPNO9
ALTER 119,126
ENDALTER
```

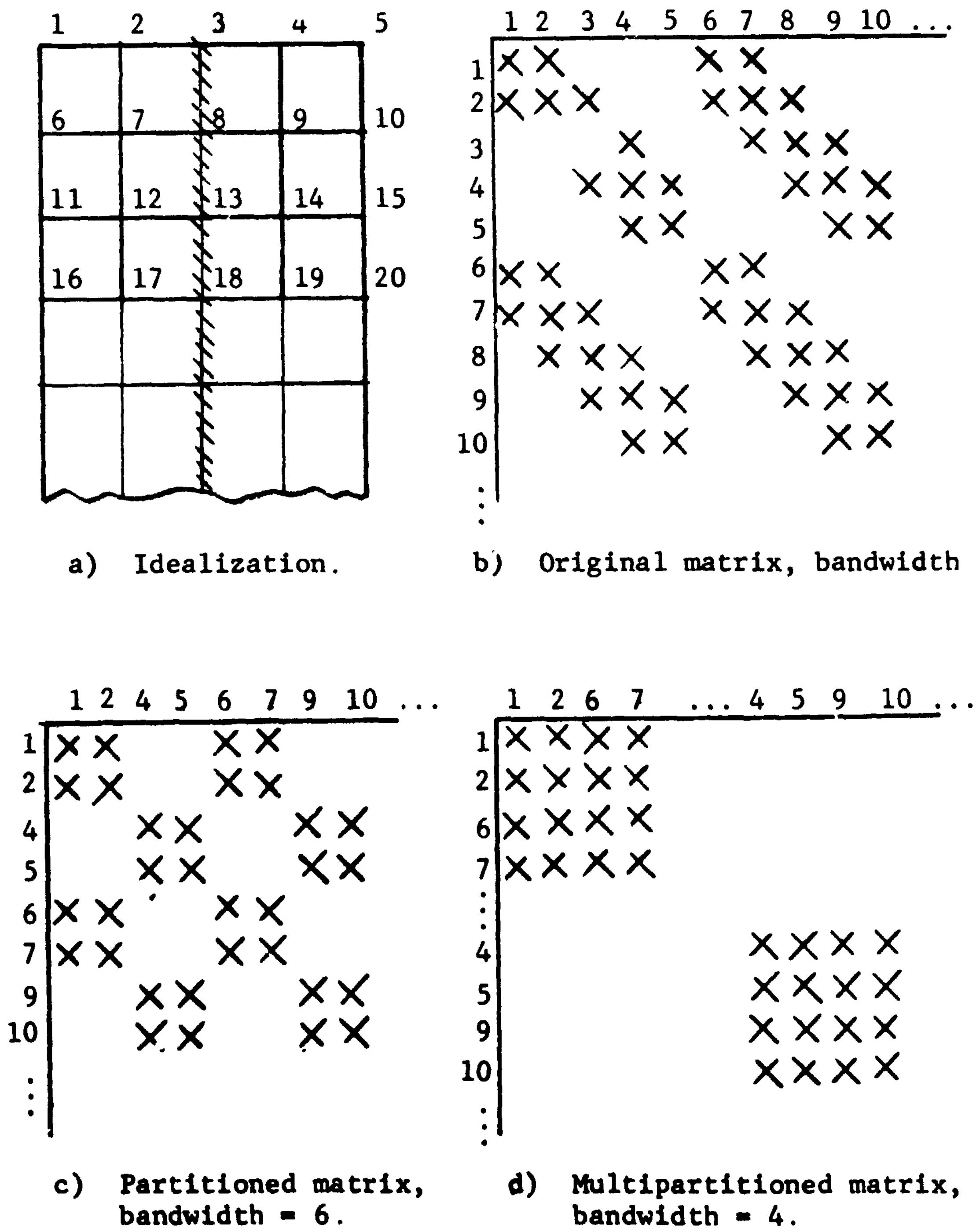
Figure 2.- Concluded.

```
ALTER 1,1
BEGIN NO.1 STATIC ANALYSIS-SERIES M1-ONE SHOT SUBSTRUCTURING $
ALTER 50
PARAM //C,N,NOP/V,N,P1=-1
PARAM //C,N,NOP/V,N,TRUE=-1 $
$ SUBNUM = NO.OF SUBSTRUCTURE TO BE CHANGED,DEFAULT=-1 CORR. TO NO CHANGE
$ NUMSUB = NO. OF SUBSTRUCTURES - ONLY USED IF SUBNUM POSITIVE
PARAM //C,N,NOP/V,Y,SUBNUM=-1 $
PARAM //C,N,NOP/V,Y,NUMSUB=1 $
PARAM //C,N,SUB/V,N,NUMS1/V,Y,NUMSUB/C,N,1 $
PARAM //C,N,SUB/V,N,SUB1/C,N,1/V,Y,SUBNUM $
PARAM //C,N,NOT/V,N,OUTP/V,Y,SUBNUM $
ALTER 54
$  FORM PARTITION VECTOR G(L,COMP)
VEC USET/V/C,N,G/C,N,L/C,N,COMP $
CHKPNT V $
$  FORM PARTITION VECTOR F(L,COMP)
VEC USET/VFLC/C,N,F/C,N,L/C,N,COMP $
CHKPNT VFLC $
PRTPARM //C,N,0 $
ALTER 75
$  PARTITION OUT L-SET
$  L-SET IS SAME FOR ALL SUBSTRUCTURES
UPARTN USET,KFF/KLLB,,KAO,/C,N,F/C,N,A/C,N,0 $
CHKPNT KLLB,KAO $
ALTER 84
JUMP LBL7 $
ALTER 96
$  PARTITION OUT L-SET
$  L-SET IS SAME FOR ALL SUBSTRUCTURES
PARTN PG,,V/PLB,,,/C,N,1/C,N,2/C,N,2 $
CHKPNT PLB $
ALTER 100,101
SSG2 USET,GM,YS,KFS,GO,,PG/,PO,PS,PL $
CHKPNT PO,PS,PL $
ALTER 102
FBS LOO,UOO,PO/UOOV $
CHKPNT UOOV $
MATPRN UOOV,PO,,,//$
MATGPR GPL,USET,SIL,UOOV//C,N,0 $
MATGPR GPL,USET,SIL,PO//C,N,0 $
ALTER 103,111
PARAM //C,N,ADD/V,N,P1/V,N,P1/C,N,1 $
ALTER 118
COND LBOUT,SUBNUM $
PARAM //C,N,SUB/V,N,P1/V,Y,NUMSUB /C,N,1 $
INPUTT1 /,,,,/C,N,-3/C,N,3 $
INPUTT1 /,,,,/C,N,-3/C,N,4 $
INPUTT1 /KLLB1,,,,/C,N,0/C,N,3 $
INPUTT1 /PLB1,,,,/C,N,0/C,N,4 $
ADD KLLB1,/KLLT/ $
ADD PLB1,/PLT/ $
LABEL FMKPL $
COND LBLSUB,SUB1 $
MPYAD KAO,GO,KLLT/KLLTX2/C,N,0 $
EQUIV KLLTX2,KLLT/TRUE $
MPYAD GO,PO,PLT/PLTX2/C,N,1 $
EQUIV PLTX2,PLT/TRUE $
PARAM //C,N,SUB/V,N,SUB1/V,N,SUB1/C,N,100 $
$ SKIP UNWANTED INFORMATION OF NON CHANGED SUBSTRUCTURES
INPUTT1 /DUM1,DUM2,,,/C,N,0/C,N,3 $
INPUTT1 /DUM3,DUM4,,,/C,N,0/C,N,4 $
JUMP LB111 $
LABEL LBLSUB $
INPUTT1 /KAO1,GO1,,,/C,N,0/C,N,3 $
INPUTT1 /GOT1,PO1,,,/C,N,0/C,N,4 $
```

Figure 3.- DMAP ALTER package for one-shot substructuring.

```
MPYAD KAO1,GO1,KLLT/KLLTX3/C,N,0 $
EQUIV KLLTX3,KLLT/TRUE $
MPYAD GOT1,PO1,PL,/PLTX3/C,N,0 $
EQUIV PLTX3,PLT/TRUE $
PARAM //C,N,ADD/V,N,SUB1/V,N,SUB1/C,N,1 $
LABEL LB111 $
PARAM //C,N,SUB/V,N,NUMS1/V,N,NUMS1/C,N,1 $
COND LBOUT,NUMS1 $
REPT FMKPL,100 $
LABEL LBOUT $
SOLVE KLLT,PLT/ULLB/C,N,1 $
CHKPNT ULLB $
MATPRN ULLB,PLT,KLLT,, // $
PARAM //C,N,NOP/V,N,NSKIP1=1 $
INPUTT1 /,,,,/C,N,-3/C,N,0/C,N,SUB $
PARAM //C,N,SUB/V,N,SUB1/C,N,1/V,Y,SUBNUM $
LABEL LBL110
INPUTT1 /USET1,UOOV1,GO4,PG1,/C,N,0/C,N,0 $
COND LBLNM,SUB1 $
JUMP LBLOO $
LABEL LBLNM $
EQUIV ULLB,ULX2/TRUE/UOOV1,UOX2/TRUE/PG1,PGX2/TRUE $
JUMP LBLNO $
LABEL LBLOO $
$ CREATE NULL MATRICES FOR NON CHANGED SUBSTRUCTURES
ADD ULLB,/ULX1/C,N,(-1.0,0.0) $
ADD ULLB,ULX1/ULX2  $
ADD UOOV1,/UOX1/C,N,(-1.0,0.0) $
ADD UOOV1,UOX1/UOX2 / $
ADD PG1,/PGX1/C,N,(-1.0,0.0) $
ADD PG1,PGX1/PGX2/ $
PARAM //C,N,ADD/V,N,NSKIP1/V,N,NSKIP1/C,N,1 $
PARAM //C,N,SUB/V,N,SUB1/V,N,SUB1/C,N,100 $
LABEL LBLNO $
SDR1 USET1,PGX2,ULX2,UOX2,,GO4,,,,,/UGV,PGG,/V,N,NSKIP1/C,N,STATICS $
CHKPNT UGV,PGG $
MATPRN UGV,PGG,,,// $
PARAM //C,N,ADD/V,N,NSKIP1/V,N,NSKIP1/C,N,1 $
PARAM //C,N,SUB/V,N,P1/V,N,P1/C,N,1 $
PARAM //C,N,ADD/V,N,SUB1/V,N,SUB1/C,N,1 $
COND LBL80,P1 $
REPT LBL110,100 $
JUMP ERROR1 $
LABEL LBL80 $
PARAM //C,N,NOT/V,N,TEST1/V,N,P1 $
COND ERROR5,TEST1 $
INPUTT1 /,,,,/C,N,-3/C,N,5 $
INPUTT1 /CASECC1,MPT1,EQEXIN1,SIL1,BGPDT1/C,N,0/C,N,5 $
INPUTT1 /PGG1,EST1,,,/C,N,0/C,N,5 $
SDR2 CASECC1,,MPT1,,EQEXIN1,SIL1,,,BGPDT1,PGG,,UGV,EST1,/OPG11,,
     OUGV11,OES11,OEF11,/C,N,STATICS $
OFP OPG11,OUGV11,OES11,OEF11,,//V,N,CARDNO $
SAVE CARDNO $
PARAM //C,N,NOP/V,N,NSKIP4=1 $
SDR1 USET,PG,ULLB,UOOV,,GO,,,,,/UGVN,PGN,/V,N,NSKIP4/C,N,STATICS $
SDR2 CASECC,CSTM,MPT,,EQEXIN,SIL,,EDT,BGPDT,PGN,,UGVN,EST,/OPG1,,
        OUGV1,OES1,OEF1,/C,N,STATICS $
OFP OPG1,OUGV1,OES1,OEF1,,//V,N,CARDNO $
SAVE CARDNO $
PRTPARM //C,N,0 $
INPUTT1 /,,,,/C,N,-3/C,N,3/C,N,TP3 $
INPUTT1 /,,,,/C,N,-3/C,N,4/C,N,TP4 $
INPUTT1 /,,,,/C,N,-3/C,N,5/C,N,TP5 $
ALTER 119,126
ENDALTER
```

Figure 3.- Concluded.



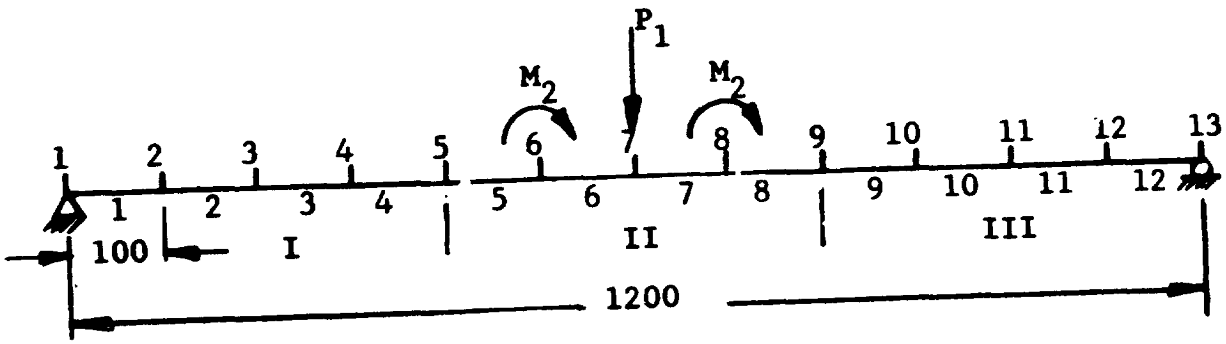
a) Idealization.

b) Original matrix, bandwidth = 7.

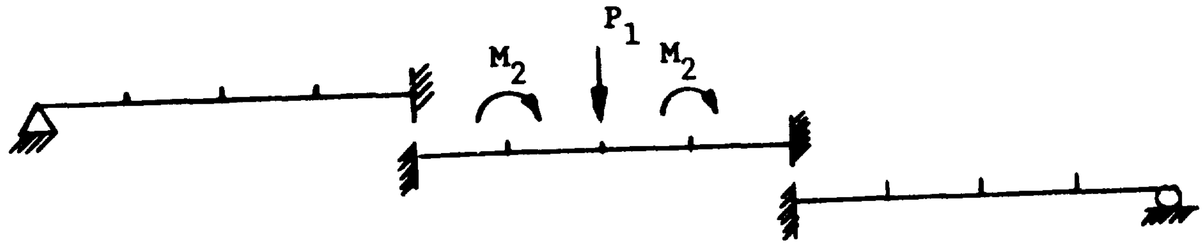
c) Partitioned matrix, bandwidth = 6.

d) Multipartitioned matrix, bandwidth = 4.

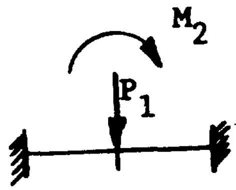
Figure 4.- Example of multipartitioning to reduce bandwidth.



a) Simply supported beam divided into three substructures.



b) Intermediate solution.



c) Change in geometry and load conditions on substructure II.

Figure 5.- Sample problem for multipartitioning and one-shot substructuring.

```
TITLE=MULTIPARTITIONING
SUBTITLE=BEAM DIVIDED INTO THREE SUBSTRUCTURES
SET 1=1 THRU 5
SET 2=5 THRU 9
SET 3=9 THRU 13
SET 4=1 THRU 4
SET 5=5 THRU 8
SET 6=9 THRU 12
DISP = ALL
STRESS = ALL
FORCE = ALL
OLOAD=ALL
SUBCASE 11
  LABEL = SUBSTRUCTURE ONE,LOAD ONE
    SPC = 1
DISP=1
OLOAD=1
STRESS=4
FORCE=4
SUBCASE 12
  LABEL = SUBSTRUCTURE ONE,LOAD TWO
    SPC = 1
DISP=1
OLOAD=1
STRESS=4
FORCE=4
SUBCASE 21
  LABEL = SUBSTRUCTURE TWO,LOAD ONE
    SPC = 2
      LOAD = 1
DISP=2
OLOAD=2
STRESS=5
FORCE=5
SUBCASE 22
  LABEL = SUBSTRUCTURE TWO,LOAD TWO
    SPC = 2
      LOAD = 2
DISP=2
OLOAD=2
STRESS=5
FORCE=5
SUBCASE 31
  LABEL = SUBSTRUCTURE THREE,LOAD ONE
    SPC = 3
DISP=3
OLOAD=3
STRESS=5
FORCE=6
SUBCASE 32
  LABEL = SUBSTRUCTURE THREE,LOAD TWO
    SPC = 3
DISP=3
OLOAD=3
STRESS=5
FORCE=6
```

Figure 6.- Example problem Case Control and Bulk Data Decks for multipartitioning.

```
BEGIN BULK
ASET    5       126     9       126
CBAR    1       100     1       2                       1.0             1
CBAR    2       100     2       3                       1.0             1
CBAR    3       100     3       4                       1.0             1
CBAR    4       100     4       5                       1.0             1
CBAR    5       100     5       6                       1.0             1
CBAR    6       100     6       7                       1.0             1
CBAR    7       100     7       8                       1.0             1
CBAR    8       100     8       9                       1.0             1
CBAR    9       100     9       10                      1.0             1
CBAR    10      100     10      11                      1.0             1
CBAR    11      100     11      12                      1.0             1
CBAR    12      100     12      13                      1.0             1
FORCE   1       7       0       1.0                     -1.0
GRDSET          0                                       0       345
GRID    1                   0.0                                 12345
GRID    2                 100.0
GRID    3                 200.0
GRID    4                 300.0
GRID    5                 400.0
GRID    6                 500.0
GRID    7                 600.0
GRID    8                 700.0
GRID    9                 800.0
GRID    10                900.0
GRID    11               1000.0
GRID    12               1100.0
GRID    13               1200.0                                 2345
MAT1    100     1.0+8           0.0
MOMENT  2       6       0       1.0                             -1.0
MOMENT  2       8       0       1.0                             -1.0
PARAM   TAPE    1
PBAR    100     100     60.0    500.0
SPC     101     1       6
SPC     103     13      16
SPC1    101     123456  2       THRU    4
SPC1    102     123456  6       THRU    8
SPC1    103     123456  10      THRU    12
SPCADD  1       102     103
SPCADD  2       103     101
SPCADD  3       101     102
ENDDATA
```

Figure 6 .- Concluded.

```
TITLE=ONE SHOT SUBSTRUCTURING
SUBTITLE=CHANGE IN SUBSTRUCTURE TWO
DISP=ALL
OLOAD=ALL
  FORCE=ALL
SUBCASE 21
 LABEL=SUBSTRUCTURE TWO,LOAD ONE
 LOAD=1
SUBCASE 22
 LABEL=SUBSTRUCTURE TWO,LOAD 2
 LOAD=2
BEGIN BULK
ASET    101     126     103     126
CBAR    101     200     101     102                     1.0
CBAR    102     200     102     103                     1.0
FORCE   1       102     0       1.0                     -1.0
GRDSET          0                                       0       345
GRID    101             0.0
GRID    102             100.0
GRID    103             200.0
MAT1    200     1.0+8           0.0
MOMENT  2       102     0       1.0     0.0     0.0     -1.0
PARAM   NUMSUB  3
PARAM   SUBNUM  2
PBAR    200     200     60.0     500.0
ENDDATA
```

Figure 7.- Example problem Case Control and Bulk Data Decks for one-shot substructuring.