Mods to MYSTRAN needed for new sparse solver

1 Overview

When MYSTRAN was coded, an Intel Math Kernel Library (MKL) sparse matrix solver, version 10.2, was installed and used for over 10 years. Over time, that solver showed it lacked robustness and, all too often, crashed with no indication of what went wrong. That sparse solver is no longer in MYSTRAN. At the time of writing of this document there is only a banded matrix solver coded with routines from LAPACK. In order for MYSTRAN to be a viable program for solving large order problems, however, it is imperative that a new sparse solver be added. Currently, there is at least one promising, Open Source, candidate under consideration. The purpose of this document is to outline where, in MYSTRAN, code for a sparse solver must be added and what matrix operations are to be coded.

2 Details

There are 14 MYSTRAN subroutines that must be addressed to upgrade all matrix solve operations. It is a simple process for anyone to see what subroutines are involved, and where code must be added for a sparse solver. For those readers of this document that prefer to jump ahead and not read details, one only has to do a text search of all of the MYSTRAN source files for the text SOLLIB (for solution library). SOLLIB is a MYSTRAN parameter and is specified in the Bulk Data File (BDF) as PARAM, SOLLIB, with value BANDED or SPARSE. The current MYSTRAN code only has detailed coding for the BANDED value of SOLLIB but does have "skeleton" code for the SPARSE value. This skeleton code only shows where code is to be added along with a short description of what is to be coded. This document fills in the details of what has to be coded.

There are 4 areas in MYSTRAN in which a sparse solver would be used:

1. In the reduction of the stiffness matrix from the original G-set to the A-set (through eliminating constraints and OMIT's). All of this is done by MYSTRAN's LINK2.f90 and the procedures that it calls.

2. Solution of the equations KLL*UL = PL for the independent degrees of freedom in the L-set in statics problems. This is done by MYSTRAN's LINK3.f90 and the procedures it calls.

3. Solution for eigenvalues by both the Inverse Power and Lanczos methods. This is done by MYSTRAN's LINK4.f90 and the procedures it calls.

4. Reduction of some matrices for the Craig-Bampton model generation procedure. This is done by MYSTRAN's LINK6.f90 and the procedures it calls.

A description of the displacement sets (G, M, N, etc) is given in the Users Reference Manual, Section 3.6. The equations involved in the reduction of the stiffness and load matrices from the original G-set to the final, independent, A-set for statics (done by MYSTRAN LINK2.f90) is shown in Appendix B to the MYSTRAN Users Referance Manual. The equations for the Craig-Bampton procedure are shown in Appendix D to the MYSTRAN Users Reference Manual. Reference to those appendices should be made in order to understand the processes in which a solver will be used, as discussed below.

There are a total of 14 subroutines that need to be modified to accomplish all of the tasks outlined in the 4 points above. However, as will be discussed later, there is an order in which the author feels they can be done in order to have a useable MYSTRAN with a sparse solver before all of the tasks are accomplished. The 14 subroutines are descriped below with the folder from the GitHub repository MYSTRAN\Source as the root directory:

Modules

PARAMS.f90: The default for BDF parameter SOLLIB is currently set to BANDED. Once a sparse solver is coded the default should be set to SPARSE

LK1\L1A-BD

BD_PARAM.f90: Currently this is completely updated to allow either BANDED or SPARSE as choices of parameter SOLLIB

LK2

REDUCE_KFF_TO_KAA.f90: This subroutine reduces matrix KFF from the F-set to the A-set and O-set and decomposes matrix KOO[1]. If there are OMIT displacement components specified in the BDF (or otherwise defined via ASET or ASET1 BDF entries) there will be a KOO partition of the KFF stiffness matrix

SOLVE_GMN.f90: If there are rigid elements (or MPC's or RBE3 elements) matrix GMN will have to be solved for from RMM*GMN = - RMN. This will involve a

[1] All of the matrices mentioned here will be stored in Compressed Row Storage (CRS) form with 1-D arrays such as I_KOO, J_KOO, KOO with the actual column and nonzero value stored in J_KOO and KOO respectively. I_KOO will have the column number where each nozero term starts in that row. All of the sparse matrix arrays needed for use in a new sparse solver are already available in the subroutines above since MYSTRAN had the earlier Intel MKL originally, which used those arrays,

decomp of RMM and a forward-backward solution (FBS) for each column of RMN.

SOLVE_GOA.f90: Matrix GOA will exist if there are OMIT components. Solve the equation KOO*GOA = -KOA for GOA. At this point KOO has already been decomposed so the sparse operation here is a FBS for the columns of KOA.

SOLVE_UO0.f90: If there are applied loads on the OMIT degrees of freedom there will be a UO0 matrix. At this point KOO has already been decomposed so the sparse operation here is a FBS for the columns of PO

LK3

LINK3.f90: Two sparse matrix operations are done in this subroutine" Decompose the KAA stiffness matrix and FBS for the A-set displacements from KAA*UA = PA

LK4

EIG_INV_PWR.f90: In eigenvalue analyses it is the L-set that is used. The L-set consists of the A-set (independent set in static analyses) along with a possible R-set. The R-set comprises SUPORT degrees of freedom. Generally, the R-set is null except for Craig-Bampton substructure eigenvalue analyses so, for normal eigenvalue analyses the L-set is the A-set. In this subroutine matrix KLL is decomposed. Sparse operations here also require a decomp of KLL abd an FBS to get an eigenvector.

EIG_LANCZOS_ARPACK.f90: This subroutine needs a re-write once a sparse version of a Lanczos eigenvalue extraction is found. Currently, the Open Source ARPACK code uses banded LAPACK operations.

EIG_SUMMARY.f90: There are no sparse matrix operations performed here, only messages written to the F06 file. This subroutine has been updated for messages related to a sparse eigenvalue method

LK6 (only needed for Craig-Bampton substructure eigenvalue analyses)

CALC_KRRcb.f90: The only matrix operation in this subroutine that uses matrix decomposition is on the KRRcb matrix which is of size RxR (R being the number of degrees of freedom in the R-set). The R-set is generally quite small so that a banded decomposition is probably all that will ever be needed. However, it is mentioned here only for completeness in the event someone wants to add sparse decomp code.

SOLVE_DLR.f90: In this subroutine solution is obtained to the equation KLL*DLR = -KLR. The operations are identical to that in several above subroutines (e/g/ SOLVE_GMN.f90)

SOLVE_PHIZL1.f90: In this subroutine solution is obtained to the equation KLL*PHIZL1 = CRS3 for PHIZL1   where CRS3 = (MLR + MLL*DLR). The operations are identical to that in several above subroutines (e/g/ SOLVE_GMN.f90)

SOLVE6_SETUP.f90: In this subroutine the only matrix operation that needs to be coded sparse is the decomposition of KLL. The operations are the same as that in several other subroutines (e.g. decomp of KAA in subroutine LINK3.f90)

In LK2 – LK6 the source files all have code for the BANDED and SPARSE solutions that look like the code snippet below:

```
IF (SOLLIB == 'BANDED  ') THEN
   ! Code for banded solution here to ...
ELSE IF (SOLLIB == 'SPARSE  ') THEN
   ! Add sparse matrix code here to ...
ELSE
   FATAL_ERR = FATAL_ERR + 1
   WRITE(ERR,9991) SUBR_NAME, SOLLIB
   WRITE(F06,9991) SUBR_NAME, SOLLIB
   CALL OUTA_HERE ( 'Y' )
ENDIF
```

Code for the BANDED solution method already exists in these snippets and is based on the well known, Open Source, **L**inear **A**lgebra **PACK**age, LAPACK. New code for the sparse solver will go in the lines below that

Although there are a lot of subroutines, addressed above, for which sparse matrix operation code must be added to complete the project there is 1 which should be addressed first in order to get a MYSTRAN version running for normal static analyses:

1. LINK3.f90: decomposition of KLL and solution for the UL displacements
2. Following this I suggest finding a sparse Lanczos method and coding that for eigenvalue analyses.
3. Then the tasks of sparse op's for MPC's and OMIT's and the Craig-Bampton model can be taken on in whatever order seems appropriate

If only #1, above, is coded for sparse operations there will still have to be temporary code to make sure that operations such as reduction of KGG for MPC’s and OMIT’s are not skipped (but done with the current banded code)