Implementing Experimental Substructuring in Abaqus

Benjamin Moldenhauer, Matthew Allen, Daniel Roettgen[1], and Brian Owens[1]

University of Wisconsin – Madison
Madison, WI 53715

[1] Sandia National Laboratories*
Albuquerque, NM 87123

ABSTRACT
In many applications, components that are difficult or inconvenient to represent as a finite element model (FEM) are instead experimentally characterized to capture their contribution to the overall dynamics of an assembly. A recent advance in experimental substructuring, the Transmission Simulator method, has proven promising for coupling an experimentally identified model to analytical FEMs. While in typical applications all experimental and FEM data is imported into MATLAB and assembled in modal coordinates, it would be preferable to perform equivalent operations in finite element software where large models are readily handled and results are easily visualized and post-processed. This work details a process for importing experimental modal models into the Dassault Systèmes® SIMULIA™ Abaqus finite element analysis software suite through an application of the Transmission Simulator method. In this approach, after decoupling the TS from the experimental subsystem in MATLAB, the result is represented as single degree-of-freedom (DOF) spring-mass oscillators in Abaqus. These are coupled to the native Abaqus FEM by implementing the substructuring constraint equations as linear equation multipoint constraints between the necessary DOF. This approach is shown to perform very well in an analytical test case and reasonably well in an experimental test case. The limiting factors appear to be the quality of measured data and curve fitting used in defining the modal properties of the experimental subsystem, the quantity and location of constraint DOF, and the selection of modes used to represent each subsystem. These determine the quality of the decoupled experimental model, which dictates how accurate the result will be after coupling it to the FEM subsystem.

Keywords: Experimental Substructuring, Transmission Simulator, Abaqus, Decoupling, Finite Element Analysis

INTRODUCTION
When creating a finite element model (FEM) of a structure, it is common practice to simplify intricate geometries and interfaces, such as removing fillets and threads and assuming bonded contacts. It is appropriate to make these generalizations when doing so reduces model complexity and solution time without significantly degrading the accuracy of the results. In cases where an intricate component cannot be simplified it may be more efficient to instead conduct a modal test on the part and capture its dynamic response experimentally. This physically based model can then be constrained to the adjacent FEMs with experimental substructuring techniques, allowing for the dynamics of the complete structure to be evaluated.

Experimental substructuring has been an active area of structural dynamics research for several decades, the highlights of which can be seen in [1]. A recent development by Allen et al. [2], the Transmission Simulator (TS) method, allows for very efficient coupling of experimental and analytical substructures with excellent accuracy. Usually the modal properties of each subsystem are passed into MATLAB, where performing the necessary calculations is trivial. However, for the uninitiated, using MATLAB to import and handle each component's data and post-process results can be cumbersome, error prone, and time consuming. The accessibility and usefulness of the TS method could be vastly improved if the experimental data could be imported into finite element software and the substructuring calculations equivalently

*Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

performed within. This would allow the analyst to leverage the capabilities of that software, where large models are managed efficiently, and extensive capabilities exist for visualizing and processing results. This paper presents a framework for performing TS method experimental substructuring within Dassault Systèmes® SIMULIA™ Abaqus finite element analysis software suite. The following sections provide a brief overview of experimental substructuring and the mathematical basis for the TS method, an overview of the process required to implement this in Abaqus, and numerical and experimental test cases that evaluate the accuracy and practicality of this method relative to what is typically done in MATLAB.

BACKGROUND & THEORY

As the scale and complexity of a design increases, its structural dynamics become progressively more challenging to simulate. An unreasonable amount of computational power or time may be necessary to directly solve complicated FEMs, and intricate physical systems can require an unrealistic number of sensors to experimentally model. These difficulties may be avoided by dividing the structure into subcomponents that can be analyzed individually and combined to reproduce the total system dynamics. The techniques that are employed to reassemble these subcomponent models are referred to as dynamic substructuring, and the term experimental substructuring is used when the subcomponents are a combination of analytical and experimental models [3].

The work detailed herein is an implementation of the TS method, which is a procedure for efficiently and accurately performing experimental substructuring. In more traditional substructuring approaches, two components are assembled by defining constraints between the physical DOF at their interface. This is extremely difficult to accomplish with experimental subcomponents as it requires rotational DOF to be measured at the interface; a tedious and error prone task. The TS method circumvents this by introducing a third component, referred to as the TS, which exists in the experimental subsystem at its interface. Instead of directly assembling the experimental and analytical subsystems, a FEM of the TS is constrained such that it is decoupled from the experimental subsystem and coupled to the analytical subsystem. The TS then acts as a distributed interface that satisfies the system constraints in terms of its own dynamics. This softens the constraints, making them less sensitive to experimental noise and less likely to induce locking, which is an unrealistic increase in stiffness at the interface [2]. This process is demonstrated in Figure 1, in which the experimental subsystem is a cantilever beam with a perpendicular beam segment attached to the tip, the TS is a FEM of that attached segment, the analytical subsystem is a FEM of a free-free beam, and the resulting model is simply a longer cantilever beam. In this scenario the TS is decoupled from the experimental subsystem, removing its physical presence from the beam tip while preserving the effects of a mass-loaded interface in the remaining cantilever. This provides an improved basis for coupling to the analytical subsystem, yielding a more accurate final assembled system. A more comprehensive discussion of the TS method and its strengths and weaknesses can be found in [2, 4, 5].

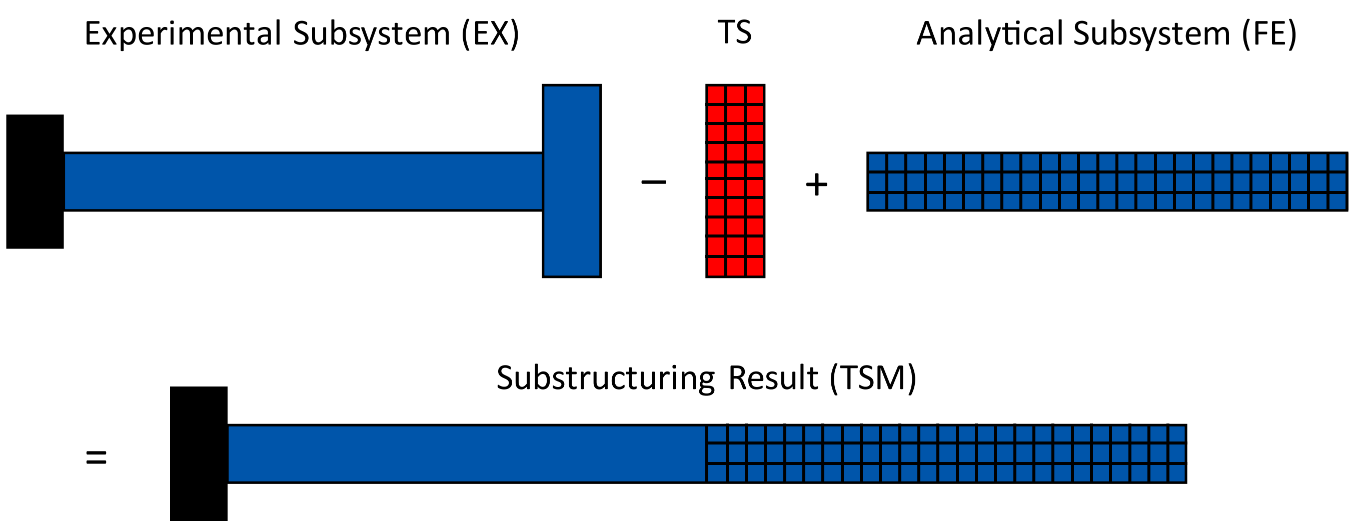

Figure 1: Visual demonstration of the TS method. The tip section of the cantilever beam is effectively removed and replaced by a second beam, yielding a longer cantilever.

Mathematically, this process is based on enforcing compatibility of displacements and force equilibrium at the interface between the subsystems. This is typically done in either a primal or dual formulation if the displacements or forces, respectively, are used as unknowns. In this work, constraints are defined in a primal formulation and used to assemble the subsystems in the modal domain with component mode synthesis (CMS), in which each subsystem is represented by its modal basis. The equation of motion for the initially uncoupled subsystems, with damping neglected for brevity, is given in Eq. (1), where $\mathbf{I}$ is an appropriately dimensioned identity matrix, $\mathbf{q}$ are vectors of modal DOF, $\backslash\boldsymbol{\omega}_{\mathrm{n}\backslash}$ are diagonal matrices of system natural frequencies, $\boldsymbol{\Phi}_{\mathrm{F}}^{\mathrm{T}}$ are the transposed mass normalized full system modeshape matrices, and $\mathbf{F}$ are vectors of external forces. Note the negative signs on the TS system terms; it is removed, or decoupled, from the experimental subsystem by coupling on a negative version of itself [2].

$$\begin{bmatrix} \mathbf{I}_{\mathrm{EX}} & 0 & 0 \\ 0 & -\mathbf{I}_{\mathrm{TS}} & 0 \\ 0 & 0 & \mathbf{I}_{\mathrm{FE}} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_{\mathrm{EX}} \\ \ddot{\mathbf{q}}_{\mathrm{TS}} \\ \ddot{\mathbf{q}}_{\mathrm{FE}} \end{bmatrix} + \begin{bmatrix} \left[\backslash\boldsymbol{\omega}^2_{\mathrm{n,EX}\backslash}\right] & 0 & 0 \\ 0 & \left[-\backslash\boldsymbol{\omega}^2_{\mathrm{n,TS}\backslash}\right] & 0 \\ 0 & 0 & \left[\backslash\boldsymbol{\omega}^2_{\mathrm{n,FE}\backslash}\right] \end{bmatrix} \begin{bmatrix} \mathbf{q}_{\mathrm{EX}} \\ \mathbf{q}_{\mathrm{TS}} \\ \mathbf{q}_{\mathrm{FE}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}^{\mathrm{T}}_{\mathrm{EX,F}} & 0 & 0 \\ 0 & -\boldsymbol{\Phi}^{\mathrm{T}}_{\mathrm{TS,F}} & 0 \\ 0 & 0 & \boldsymbol{\Phi}^{\mathrm{T}}_{\mathrm{FE,F}} \end{bmatrix} \begin{bmatrix} \mathbf{F}_{\mathrm{EX}} \\ \mathbf{F}_{\mathrm{TS}} \\ \mathbf{F}_{\mathrm{FE}} \end{bmatrix} \tag{1}$$

The primal formulation constraints are formed by enforcing the displacements to be equal at the interface, given in matrix form in Eq. (2), where $\mathbf{x}$ are vectors of physical DOF.

$$\begin{bmatrix} 1 & -1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathrm{EX}} \\ \mathbf{x}_{\mathrm{TS}} \\ \mathbf{x}_{\mathrm{FE}} \end{bmatrix} = [0] \tag{2}$$

Casting this to the modal domain with $\mathbf{x} = \boldsymbol{\Phi}\mathbf{q}$ yields Eq. (3), where $\boldsymbol{\Phi}$ are the modeshapes of the subsystems partitioned to only the DOF that are being constrained.

$$\begin{bmatrix} \boldsymbol{\Phi}_{\mathrm{EX}} & -\boldsymbol{\Phi}_{\mathrm{TS}} & 0 \\ 0 & -\boldsymbol{\Phi}_{\mathrm{TS}} & \boldsymbol{\Phi}_{\mathrm{FE}} \end{bmatrix} \begin{bmatrix} \mathbf{q}_{\mathrm{EX}} \\ \mathbf{q}_{\mathrm{TS}} \\ \mathbf{q}_{\mathrm{FE}} \end{bmatrix} = [0] \tag{3}$$

Pre-multiplying by the pseudo inverse of the TS modes, as in Eq. (4), weakens the constraints by casting them to a least squares problem between the TS modal coordinates and their orthogonal projection onto the other subsystem modal bases.

$$\begin{bmatrix} \boldsymbol{\Phi}^{\dagger}_{\mathrm{TS}} & 0 \\ 0 & \boldsymbol{\Phi}^{\dagger}_{\mathrm{TS}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Phi}_{\mathrm{EX}} & -\boldsymbol{\Phi}_{\mathrm{TS}} & 0 \\ 0 & -\boldsymbol{\Phi}_{\mathrm{TS}} & \boldsymbol{\Phi}_{\mathrm{FE}} \end{bmatrix} \begin{bmatrix} \mathbf{q}_{\mathrm{EX}} \\ \mathbf{q}_{\mathrm{TS}} \\ \mathbf{q}_{\mathrm{FE}} \end{bmatrix} = [0] \tag{4}$$

The constraint equations between the subsystems, $\mathbf{B}$, then become as given in Eq. (5).

$$\mathbf{B} = \begin{bmatrix} \boldsymbol{\Phi}^{\dagger}_{\mathrm{TS}}\boldsymbol{\Phi}_{\mathrm{EX}} & -\mathbf{I} & 0 \\ 0 & -\mathbf{I} & \boldsymbol{\Phi}^{\dagger}_{\mathrm{TS}}\boldsymbol{\Phi}_{\mathrm{FE}} \end{bmatrix} \tag{5}$$

The subsystem matrices are then assembled using a matrix $\mathbf{L}$, defined as the nullspace of $\mathbf{B}$, or $\mathbf{L} = \mathrm{null}(\mathbf{B})$, in Eq. (6).

$$\widehat{\mathbf{M}} = \mathbf{L}^{\mathrm{T}} \begin{bmatrix} \mathbf{I}_{\mathrm{EX}} & 0 & 0 \\ 0 & -\mathbf{I}_{\mathrm{TS}} & 0 \\ 0 & 0 & \mathbf{I}_{\mathrm{FE}} \end{bmatrix} \mathbf{L} \quad ; \quad \widehat{\mathbf{K}} = \mathbf{L}^{\mathrm{T}} \begin{bmatrix} \left[\backslash\boldsymbol{\omega}^{\,2}_{\mathrm{n,EX}\backslash}\right] & 0 & 0 \\ 0 & \left[-\backslash\boldsymbol{\omega}^2_{\mathrm{n,TS}\backslash}\right] & 0 \\ 0 & 0 & \left[\backslash\boldsymbol{\omega}^2_{\mathrm{n,FE}\backslash}\right] \end{bmatrix} \mathbf{L} \tag{6}$$

After computing the eigenvectors, $\boldsymbol{\Phi}_{\mathrm{CON}}$, and eigenvalues, $\boldsymbol{\lambda}_{\mathrm{CON}}$, of the constrained system, the modal properties of the assembled structure can be found as in Eq. (7). While the square root of the eigenvalues are the natural frequencies, $\boldsymbol{\omega}_{\mathrm{n,TSM}}$, the constrained system eigenvectors must be transformed back to the respective subsystem coordinates to form the modeshapes, $\boldsymbol{\Phi}_{\mathrm{TSM}}$; a more thorough derivation and explanation of this process, including methods for handling damping, can be found in [1, 2].

$$\left[\widehat{\mathbf{K}} - \boldsymbol{\lambda}_{\mathrm{CON}}\widehat{\mathbf{M}}\right]\boldsymbol{\Phi}_{\mathrm{CON}} = 0 \quad ; \quad \boldsymbol{\Phi}_{\mathrm{TSM}} = \begin{bmatrix} \boldsymbol{\Phi}_{\mathrm{EX,F}} & 0 & 0 \\ 0 & \boldsymbol{\Phi}_{\mathrm{TS,F}} & 0 \\ 0 & 0 & \boldsymbol{\Phi}_{\mathrm{FE,F}} \end{bmatrix} \mathbf{L}\boldsymbol{\Phi}_{\mathrm{CON}} \quad ; \quad \boldsymbol{\omega}_{\mathrm{n,TSM}} = \sqrt{\boldsymbol{\lambda}_{\mathrm{CON}}} \tag{7}$$

Equations (1)-(7) define the standard procedure for implementing the TS method. This is usually done in MATLAB, as the calculations are then trivial to complete. However, this process cannot be directly implemented in Abaqus as its eigensolvers, Lanczos, subspace iteration and automatic multi-level substructuring, are designed to accept FEM matrices that are always symmetric and positive definite or positive semidefinite, for mass and stiffness respectively [6]. To decouple the TS from the experimental subsystem, it is effectively given negative mass and stiffness, making both negative definite and incompatible with the Abaqus eigensolvers. To bypass this, this work divides the simultaneous decoupling and coupling operations into two separate computations. The TS is first decoupled from the experimental subsystem in MATLAB, and the resulting modal model is imported into Abaqus and coupled to the analytical subsystem.

To decouple the TS from the experimental subsystem in a standalone operation, the constraint equation matrix in Eq. (5) and the subsystem matrices in Eq. (6) are simply truncated to the terms enforcing compatibility between the TS and the experimental subsystem, as given in Eqs. (8)-(9).

$$\mathbf{B}_{\mathrm{D}} = \begin{bmatrix} \mathbf{\Phi}_{\mathrm{TS}}^{\dagger}\mathbf{\Phi}_{\mathrm{EX}} & -\mathbf{I} \end{bmatrix} \quad ; \quad \mathbf{L}_{\mathrm{D}} = \mathrm{null}(\mathbf{B}_{\mathrm{D}}) \tag{8}$$

$$\widehat{\mathbf{M}}_{\mathrm{D}} = \mathbf{L}_{\mathrm{D}}^{\mathrm{T}} \begin{bmatrix} \mathbf{I}_{\mathrm{EX}} & 0 \\ 0 & -\mathbf{I}_{\mathrm{TS}} \end{bmatrix} \mathbf{L}_{\mathrm{D}} \quad ; \quad \widehat{\mathbf{K}}_{\mathrm{D}} = \mathbf{L}_{\mathrm{D}}^{\mathrm{T}} \begin{bmatrix} \left[{}^{\backslash}\boldsymbol{\omega}_{\mathrm{n,EX}}^{2}{}_{\backslash} \right] & 0 \\ 0 & \left[-{}^{\backslash}\boldsymbol{\omega}_{\mathrm{n,TS}}^{2}{}_{\backslash} \right] \end{bmatrix} \mathbf{L}_{\mathrm{D}} \tag{9}$$

Typically, removing the TS from the experimental subsystem yields areas of residual negative mass and/or stiffness in the resultant decoupled system matrices, $\widehat{\mathbf{M}}_{\mathrm{D}}$ and $\widehat{\mathbf{K}}_{\mathrm{D}}$, making them not positive definite or positive semidefinite, respectively, and thus incompatible with Abaqus. A method for correcting this is to essentially add a set of point masses to the system to make the mass matrix positive definite [7, 8]. While not done in the authors' prior works, one can similarly add a set of grounded springs to the stiffness matrix to make it positive semidefinite. For the mass matrix, and similarly for the stiffness matrix, this is done by first computing its eigensolution and creating a vector corresponding to the amount of mass to add to each eigenvector according to Eq. (10), where $\boldsymbol{\lambda}_k$ are the eigenvalues and ϵ is a small, nonzero value.

$$\boldsymbol{\Delta\lambda}_k = \begin{cases} 0 & \boldsymbol{\lambda}_k > 0 \\ -\boldsymbol{\lambda}_k + \epsilon & \boldsymbol{\lambda}_k \leq 0 \end{cases} \tag{10}$$

This is then put into diagonal matrix, as in Eq. (11), where the only nonzero entries will be negative eigenvalues with a flipped sign and a numerically small amount added.

$$\widehat{\mathbf{\Lambda}} = \left[{}^{\backslash}\boldsymbol{\Delta\lambda}_{\backslash} \right] \tag{11}$$

The modeshapes corresponding to the negative eigenvalues, $\boldsymbol{\phi}_k$, are then collected into a matrix $\boldsymbol{\phi}_{\mathrm{M_D}}$, as in Eq. (12).

$$\boldsymbol{\phi}_{\mathrm{M_D}} = [\boldsymbol{\phi}_k \quad \cdots \quad \boldsymbol{\phi}_n] \tag{12}$$

The mass matrix is then made positive definite in Eq. (13), which essentially adds to the negative eigenvalues to make them equal to ϵ.

$$\widehat{\mathbf{M}}_{\mathrm{D}} = \widehat{\mathbf{M}}_{\mathrm{D}} + \boldsymbol{\phi}_{\mathrm{M_D}}^{\mathrm{T}}\, \widehat{\mathbf{\Lambda}}\, \boldsymbol{\phi}_{\mathrm{M_D}} \tag{13}$$

Ideally, very little mass would need to be added to make the matrix positive definite. This amount can be quantified by computing the ratio of the matrix norms as given in Eq. (14).

$$n_{\mathrm{ratio}} = \frac{\left\| \boldsymbol{\phi}_{\mathrm{M_D}}^{\mathrm{T}}\, \widehat{\mathbf{\Lambda}}\, \boldsymbol{\phi}_{\mathrm{M_D}} \right\|}{\left\| \widehat{\mathbf{M}}_{\mathrm{D}} \right\|} \tag{14}$$

The modal properties of the decoupled system, $\mathbf{\Phi}_{\mathrm{D}}$ and $\boldsymbol{\omega}_{\mathrm{n,D}}$, are then found by transforming the eigensolution of the corrected mass and stiffness matrices as in Eq. (15), where $\mathbf{\Phi}_{\mathrm{EX,F}}$ and $\mathbf{\Phi}_{\mathrm{TS,F}}$ are matrices of the subsystem basis modeshapes at all DOF. If this continues to yield negative eigenvalues, they must simply be removed from the model.

$$\left[\widehat{\mathbf{K}}_{\mathrm{D}} - \boldsymbol{\lambda}_{\mathrm{CON,D}} \widehat{\mathbf{M}}_{D} \right] \mathbf{\Phi}_{\mathrm{CON},D} = [0] \quad ; \quad \mathbf{\Phi}_{\mathrm{D}} = \begin{bmatrix} \mathbf{\Phi}_{\mathrm{EX,F}} & 0 \\ 0 & \mathbf{\Phi}_{\mathrm{TS,F}} \end{bmatrix} \mathbf{L}_{\mathrm{D}} \mathbf{\Phi}_{\mathrm{CON,D}} \quad ; \quad \boldsymbol{\omega}_{\mathrm{n,D}} = \sqrt{\boldsymbol{\lambda}_{\mathrm{CON,D}}} \tag{15}$$

The decoupled model may then be imported into Abaqus as a set of modal DOF with unit mass and a stiffness equal to the square of the natural frequency. To constrain these to the analytical subcomponent in Abaqus, the compatibility condition is formed as shown in Eq. (16), where $\mathbf{x}_{\mathrm{FE}}$ corresponds to a set of physical DOF in the Abaqus FEM to constrain to.

$$\mathbf{\Phi}_{\mathrm{TS}}^{\dagger}[\mathbf{\Phi}_{\mathrm{D}} \quad \mathbf{I}]\begin{bmatrix}\mathbf{q}_{\mathrm{D}} \\ \mathbf{x}_{\mathrm{FE}}\end{bmatrix} = [0] \tag{16}$$

The constraint equations are then as given in Eq. (17), which can be implemented in Abaqus as a set of linear equation multipoint constraints.

$$\mathbf{B}_{\mathrm{ABQ}} = \left[\mathbf{\Phi}_{\mathrm{TS}}^{\dagger}\mathbf{\Phi}_{\mathrm{D}} \quad \mathbf{\Phi}_{\mathrm{TS}}^{\dagger}\right] \tag{17}$$

IMPLEMENTATION

The general process for implementing the TS method in Abaqus as presented in the previous section is detailed in the following steps. This procedure will require the use of MATLAB and Abaqus and assumes that work has already been completed in designing the subcomponents, measuring experimental data, and creating the necessary FEMs. This data is then imported into MATLAB where constraint DOF are identified and the TS is decoupled from the experimental subsystem. The resultant decoupled model is then imported into Abaqus and constrained to the analytical subsystem.

1. Gather Subsystem Data and Import into MATLAB

The substructuring process begins with designing the experimental subsystem, the TS, and the analytical subsystem; several guidelines for creating a suitable TS are given in [5]. From a modal test of the experimental component and a FEM of the TS, the linear natural frequencies and modeshapes of both may be determined; damping is optional. This data is then imported into MATLAB, along with node locations and DOF directions for every subsystem, including the analytical component.

2. Identify Constraint DOF

In order to define the substructuring constraint equations, the DOF to be constrained between each subsystem must be determined. After ensuring that all components are defined within the same global coordinate system, or performing any transformations required to make them so, this can be done by first locating node pairs between the components and then matching any common DOF directions at each pairing. Typically, the constraints are defined in terms of the experimental subsystem DOF, as in pairings are only found between it and the other subsystems. However, since the TS and analytical component are FEMs, it is possible to define the constraints at every viable node pair between them. While this may yield improved results, it can also drastically increase the number of terms in each constraint equation. The Abaqus documentation advises against this, as implementing long equations can severely degrade solver performance [6].

3. Decouple the TS from the Experimental Subsystem

To implement Eqs. (8)-(15) in MATLAB, one must first determine a suitable modal basis for the experimental subsystem and the TS. This is arguably the most critical step in the substructuring process as the resulting model can be incredibly sensitive to what modes are used in its creation. In general, a suitable modal basis is one that adequately spans the system dynamics in the frequency range of interest and, for the TS, it is critical that it does not contain linearly dependent modeshapes when partitioned to the test DOF in order to avoid numerical ill-conditioning in its pseudo inverse. Several techniques and metrics that assist in selecting appropriate modes are given in [4, 8].

4. Form Constraint Equations for use in Abaqus

The decoupled model computed in the previous step is then coupled to the Abaqus analytical model as in Eq. (16). The constraint equations, shown in matrix form in Eq. (17), are computed with the pseudo inverse of the TS modal basis and the modeshapes of the decoupled model. These can be implemented in Abaqus as linear equation multipoint constraints as each column corresponds to a constraint DOF, each row an equation, and each entry a term in that equation. However, Abaqus enforces linear equations by eliminating the first term DOF from the model, meaning that it cannot appear in any other equation [6]. Since Eq. (17) likely produces a fully populated matrix, these constraint equations must be altered, such as putting them into reduced row echelon form (RREF), which removes the pivot, or leading nonzero term in that row, from all other rows. If the built-in MATLAB function for RREF is used, it may use a numerically zero term as a pivot and produce arbitrarily large terms in the other rows. This destroys the conditioning of those equations, which can cause errors in Abaqus later. To avoid this, the columns of the constraint matrix should be reordered such that the leading diagonal does not contain any arbitrarily small terms.

5. Write Auxiliary Abaqus Input File

To complete the substructuring procedure, the decoupled model and constraint equations are imported into Abaqus by writing a text file to be processed with the input file for the analytical subsystem FEM. This auxiliary input file defines the decoupled model in terms of its modal DOF by generating an appropriate quantity of nodes, assigning each a point mass element with unit magnitude, attaching a grounded spring element with stiffness equal to the square of each natural frequency, and constraining out the other five DOF to make them single DOF oscillators. The constraint equations are then written as linear equations relating the motion of these decoupled model DOF to the physical FEM DOF. If the constraint equation matrix was reordered in the previous step, care must be taken to ensure that each equation coefficient is associated with the correct DOF. A command must then be placed in the analytical subsystem input file instructing Abaqus to include the auxiliary input file when processing that analysis. To enable a painless application of this process, Appendix A contains a MATLAB script that, given the necessary data, writes the auxiliary input file in the correct Abaqus syntax.

NUMERICAL CASE STUDY

To evaluate the basic usability and practicality of implementing the proposed substructuring method, a numerical case study was conducted. To facilitate ease of application and interpretation, a system of simple 2D beam FEMs were chosen for the subsystems as presented in Figure 2. The experimental subsystem is a 1 [m] long cantilever, the TS is a 0.2 [m] long free-free beam, and the analytical subsystem is a 1 [m] long free-free beam. A FEM of each was constructed in Abaqus from 500 B21 two node linear beam elements with a 0.01 [m] square cross section, 70 [GPa] elastic modulus, Poisson's Ratio of 0.33, and 2700 [kg/m^3] density. The experimental subsystem modeshapes were truncated to only two translational DOF every 0.1 [m] along the beam to simulate a sparse sensor placement. The subsystems were assembled as shown in Figure 2, where the vertical and horizontal DOF of the last three nodes at the tip of the cantilever were constrained to the six matching DOF on the TS and analytical subsystem. In this scenario, the tip of the cantilever is effectively removed and replaced by a longer beam section, yielding in a longer cantilever. Thus, the truth data for this setup is a 1.8 [m] cantilever beam composed of 900 elements with the properties stated above.

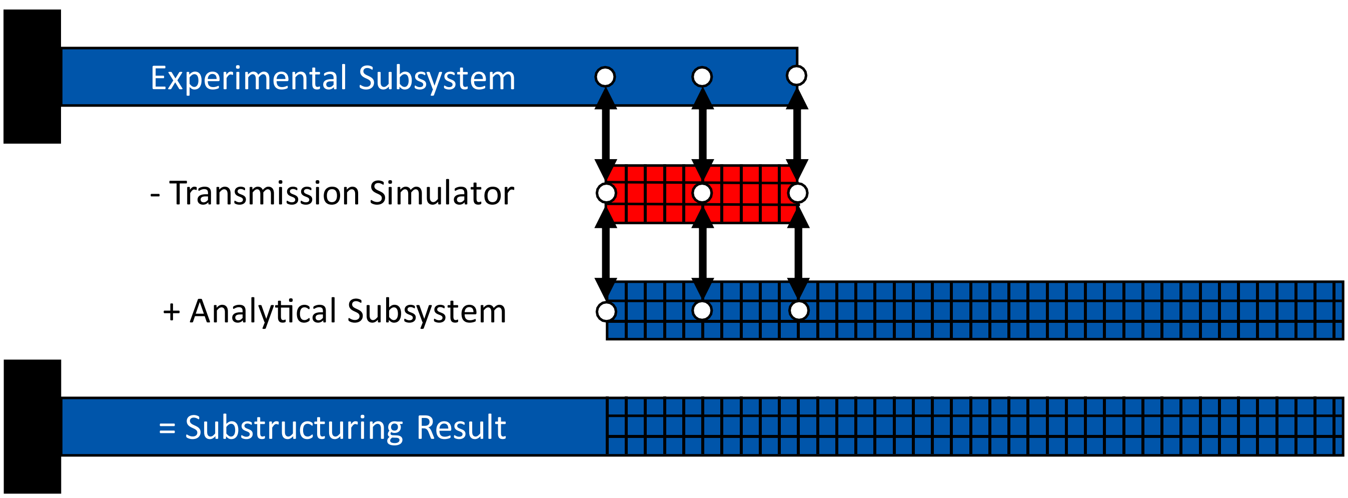

Figure 2: Subsystem layout for the simple beam numerical case study.

After importing the subsystem data into MATLAB and locating the indices of the DOF to constrain in each, the modal bases for the TS and experimental subsystem are formed. In this case study, the first step was to select TS modes based on examining their modal assurance criterion (MAC) at the constraint DOF. A MAC is essentially a measure of how similar a pair of modes are and was used as an indication of linear independence; a low MAC value signifies linear independence and a high value represents linear dependence. The left side of Figure 3 depicts a plot of the self MAC of the TS modes, where large off-diagonal terms indicate dependent modes. This shows that modes 5, 6, 7, 9, 10, 12 and beyond are aliased versions of modes 1, 2, and 4, which are the rigid body modes (RBM) and first flexural mode in the transverse direction. Modes 3, 8, and 11 are the RBM and first two flexural modes in the axial direction; the aliased copies of these are beyond the twelve modes shown in the plot. As there are three constraint DOF in each of these directions, transverse and axial, only three modes in each direction can be clearly resolved. Thus, the MAC indicates that modes 1, 2, 3, 4, 8, and 11 of the TS are suitable candidates for its modal basis.

For the experimental subsystem, the standard substructuring procedure would be to use all available modes in its modal basis. However, in this case study, it was observed that as more modes were included in the experimental subsystem, a larger correction was required to eliminate negative eigenvalues from the decoupled model, as computed in Eqs. (8)-(14). This negatively impacted the accuracy of the subsequent assembly of the corrected decoupled model and the analytical subsystem, with larger corrections yielding a worse final substructuring result. Therefore, in this work modes were removed from both the TS and experimental subsystem modal bases in order to minimize the amount of correction needed for the decoupled model. This resulted in the experimental subsystem containing modes 1-11, 15, 20, and 24, and mode 11 being removed from the TS, which reduced the mass matrix correction ratio, Eq. (14), from 0.96 to 0.03. A possible explanation for this is given on the right side of Figure 3, which illustrates the self MAC of the experimental subsystem modes at the constraint DOF. This shows that the transverse bending modes begin to significantly alias past mode 11; modes 8, 15, 20, and 24 are axial modes and show a similar trend beyond what is shown in the plot. Thus, contrary to standard substructuring practice, it seems that aliased experimental subsystem modes decrease substructuring accuracy in the separated approach proposed in this work.

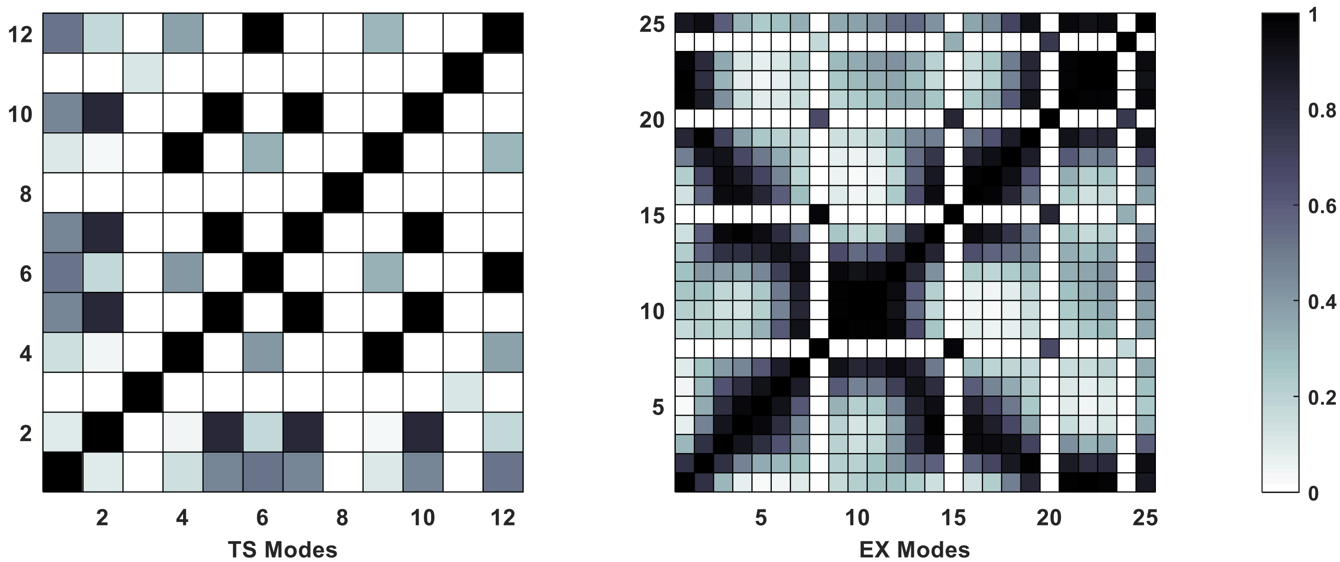

Figure 3: Self MAC of TS and EX modes at the constraint DOF showing where each begins to alias.

With the TS and experimental subsystem modal bases determined above, the resultant decoupled model is constrained to the analytical subsystem as in Eqs. (16)-(17). This is completed in Abaqus by generating the text file to be included with the input file for the analytical subsystem FEM. Appendix B contains a truncated version of this auxiliary input file as written by the script in Appendix A as an example of what should be expected when this proposed procedure is properly applied.

In the following results, the truth data is compared to three different methods for completing the substructuring computations. 'Control' is the result if the standard substructuring procedure presented in Eqs. (1)-(7) is implemented in MATLAB, 'MATLAB' is the result if the analytical subsystem is imported into MATLAB and constrained to the decoupled model in modal coordinates, and 'Abaqus' is the result of implementing the method proposed in this work where the decoupled model is imported into Abaqus and constrained to the FEM there.

A MAC between the modeshapes from the MATLAB and Abaqus results, displayed in Figure 4, shows that the two methods yield essentially identical models, as the MAC matrix is nearly symmetric. Natural frequencies are also effectively equivalent, as there is less than 0.003% difference in the prediction from each. A similar result is found when comparing the Abaqus and Control results, where the modeshapes are indistinguishable and natural frequencies estimates are within 0.4%.

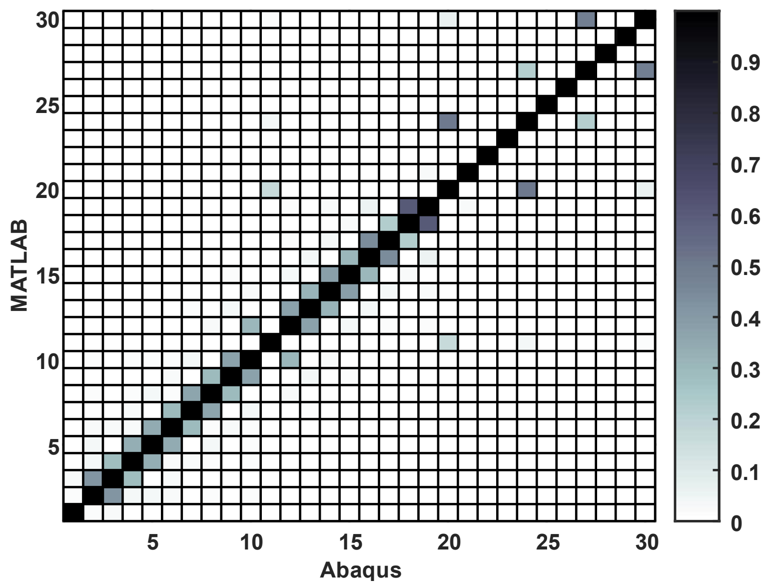

Figure 4: MAC between the MATLAB and Abaqus results, showing that they are identical.

To easily compare the substructuring predictions to the truth model, frequency response functions (FRFs) at the beam tip were generated from the modeshapes, natural frequencies, and an assumed constant damping ratio of 0.5%. The beam motion in the transverse direction is given in Figure 5, and in the axial direction in Figure 6. These show that, while the results from each of the three substructuring methods are functionally identical, they only agree with the truth data up to 2000 [Hz] in the transverse direction and 9000 [Hz] in the axial direction. This is due to the experimental subsystem modal basis containing transverse modes up to 2050 [Hz] and axial modes up to 8910 [Hz]; there is no dynamic information past these from which the substructuring methods can accurately predict resultant behavior.

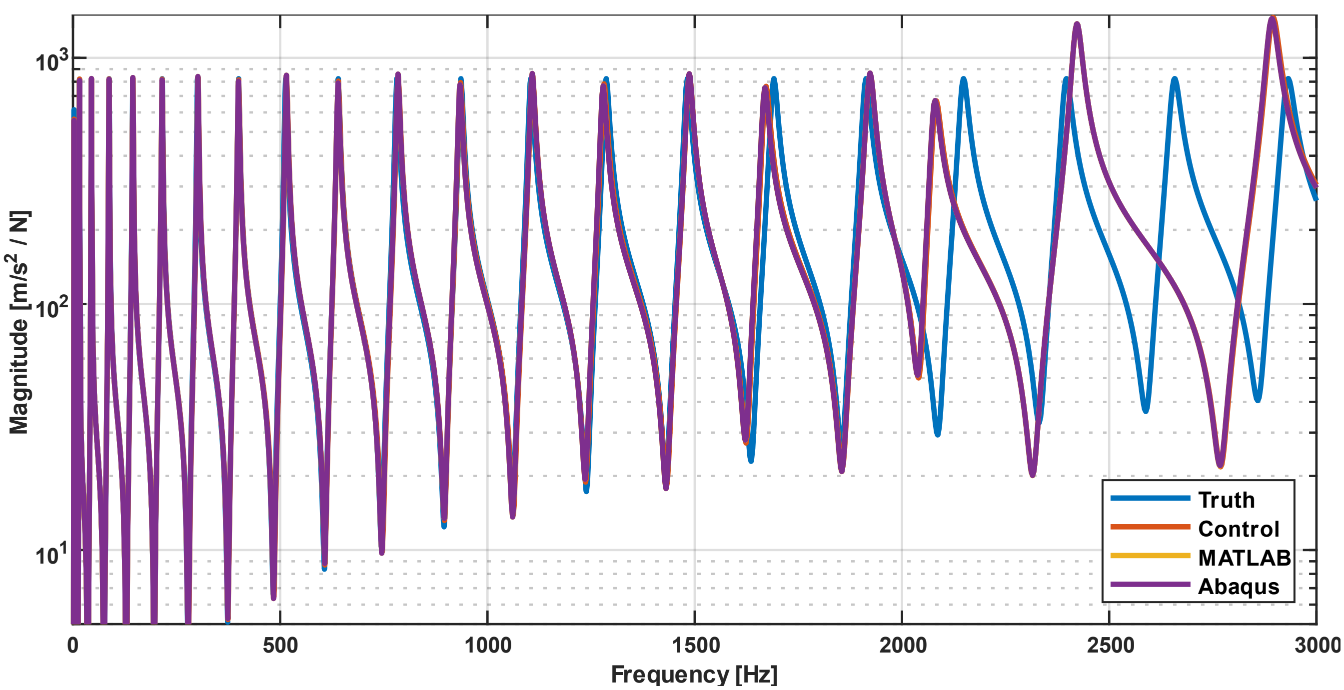

Figure 5: FRFs of the beam tip motion in the transverse direction

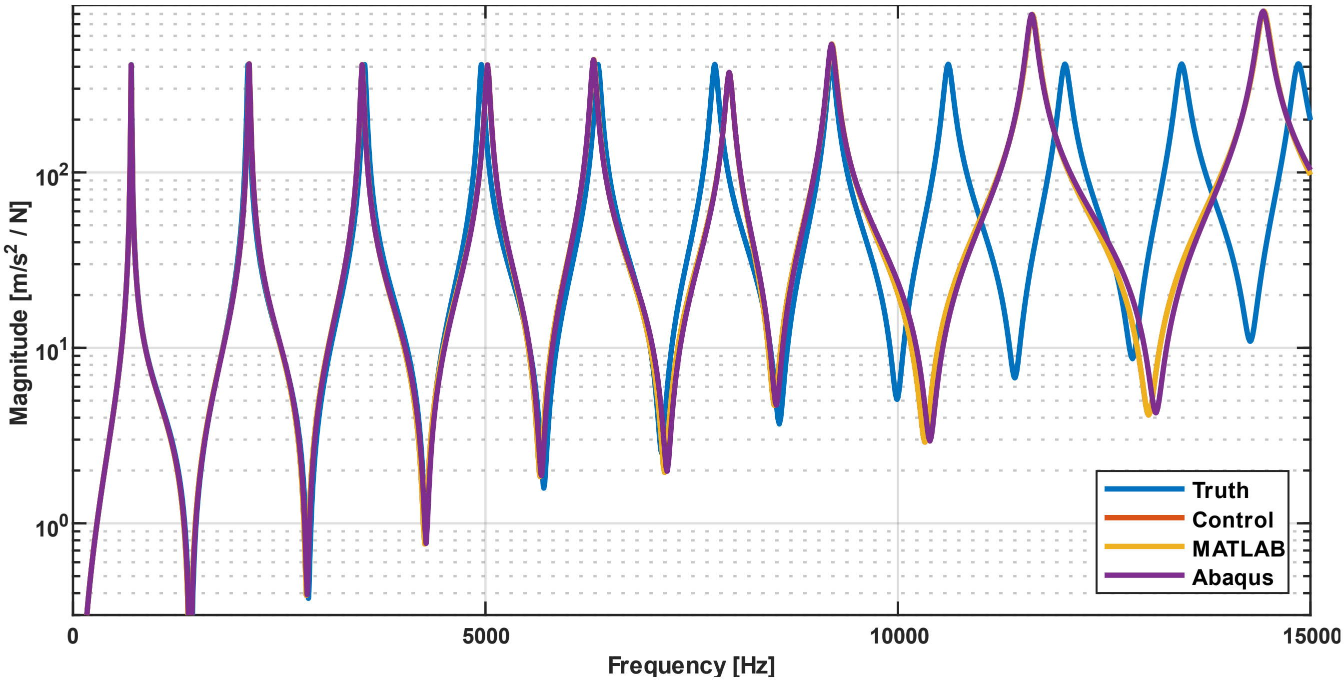

Figure 6: FRFs of beam tip motion in the axial direction

As a final check, the MAC between the truth model and Abaqus substructuring results is shown in Figure 7. The diagonal up to mode 18 shows perfect agreement, with small off diagonal terms simply due to subsequent beam modes being inherently somewhat similar. While the results seem to diverge past mode 18, this is what was previously observed in the beam tip FRFs; the transverse modes are accurate up to 2000 [Hz], which consists of the first 18 modes. Past this point in the MAC, several mode pairs corresponding to the axial modes show high correlation which is also in agreement with the axial FRF.

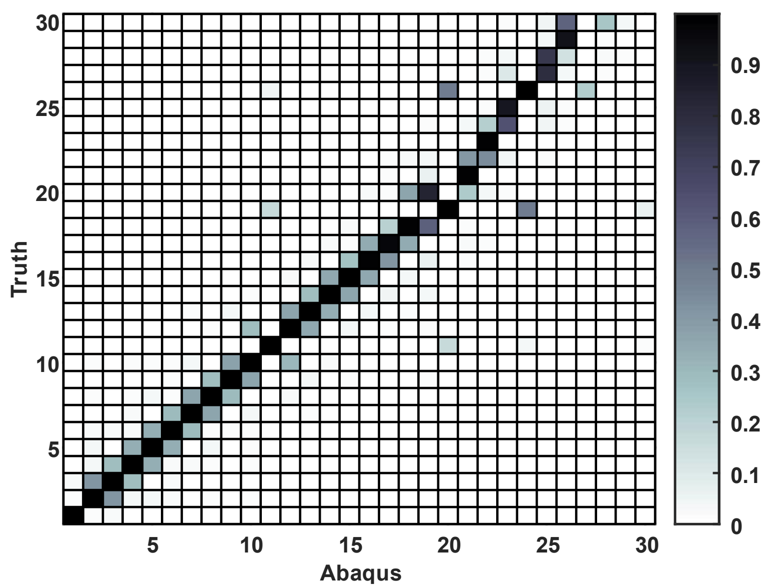

Figure 7: MAC between truth model and Abaqus substructuring results

This numerical case study demonstrates that the TS method can be implemented in Abaqus to very great effect; the excellent level of agreement between all substructuring methods and the truth model is very encouraging. Assuming the standard substructuring method yields the optimal model from the selected modal bases, the proposed separated method approaches the same result by minimizing the amount of correction needed in the decoupled model. Although the assembled model is accurate throughout the entire frequency range of the experimental modal basis, this is largely limited by the selected constraint DOF. If more DOF were included in the interface region of the experimental subsystem, higher order modes could be included in the TS modal basis without aliasing, allowing the TS to be more flexible and accommodate higher frequency experimental subsystem dynamics at the interface. This shows that, while the strengths of the TS method arise from satisfying the subsystem constraints via the TS, this also limits its applicable dynamic range to what can be projected through the TS modal basis.

EXPERIMENTAL TEST CASE

The proposed Abaqus substructuring procedure was then applied to an experimental test case where it may be compared to the other methods in a more realistic setting. The subsystems used in this test case are shown in Figure 8, where the experimental subsystem is an electrodynamic shaker with a three sided half cube adapter mounted to the armature, the TS is the half cube adapter, and the analytical subsystem is the half cube with an attached cantilever beam. See [9] for a more detailed description of these subsystems. The resulting substructured model is a prediction of how the dynamics of the shaker and adapter will change due to the addition of the beam. In general, the analytical subsystem could be a FEM of any proposed test setup and the substructured model can be used to identify any problematic dynamics of the system prior to a test taking place. This was previously implemented in [9] with the standard TS method, where all subcomponent data was imported into MATLAB and the constraints were applied simultaneously. As in the numerical test case, that method, deemed the ‘Control’ result, will here be compared against coupling the decoupled model to the analytical subsystem in either MATLAB or Abaqus, again referred to as the ‘MATLAB’ and ‘Abaqus’ results.

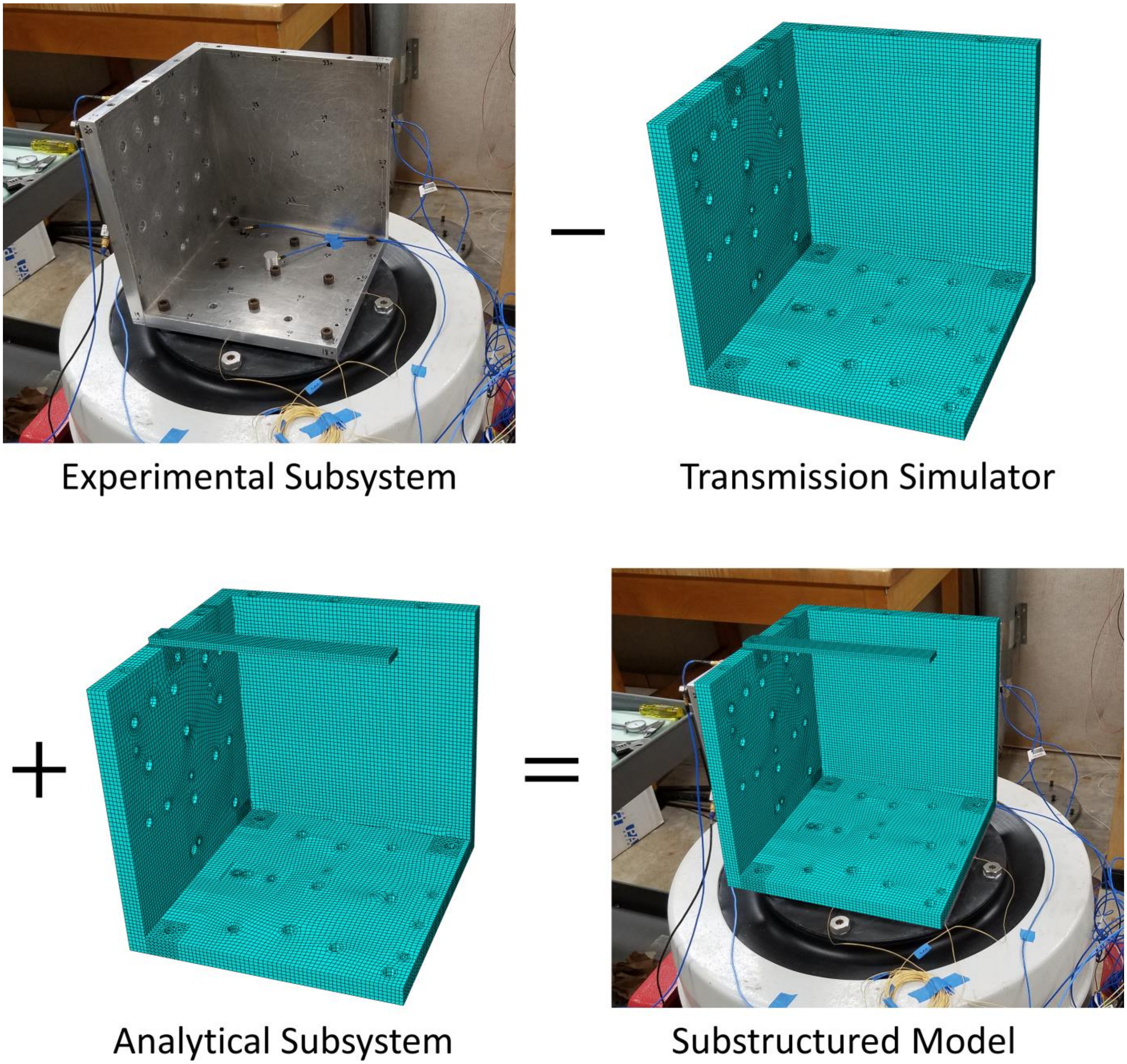

Figure 8: Experimental Test Case Subsystems

The subsystems are constrained at every measurement point utilized in characterizing the experimental subsystem. The location and direction of these 51 DOF, shown in Figure 9, were chosen with Effective Independence [10] to maximize the linear independence of the first 35 modes of the TS. This optimizes the test setup while providing the best possible array of modes from which the TS modal basis can be selected. Thus, by minimizing the amount of correction needed in decoupling the TS from the experimental subsystem, the first 22 modes of the TS, up to 3400 [Hz], and the first 26 modes of the experimental subsystem, up to 3875 [Hz], were selected for the modal bases of each. For the decoupled model, these require a correction of 0.22% to the stiffness matrix and no correction to the mass matrix.

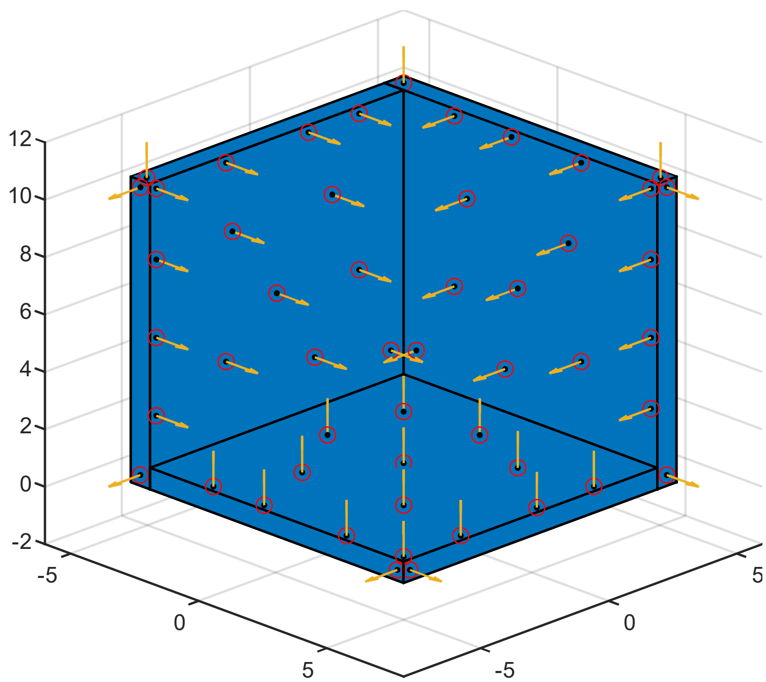

Figure 9: Location and direction of experimental test case constraint DOF

The results of the different substructuring methods are presented as FRF and MAC plots in Figure 10. As damping was not computed in any of the substructuring computations, a damping ratio of 0.1% was assumed when generating the FRFs at each measurement point, which are then averaged into the single curve shown. Each row of plots corresponds to an implementation of substructuring, with the top showing the 'Control' result, the middle the 'MATLAB' result, and the bottom displays the 'Abaqus' model. In each of these FRF plots, the substructured model is shown relative to a set of truth data measured from the physical system with a cantilever beam mounted on the half cube and shaker. To the right of the FRFs, MAC plots between the substructured and truth model modeshapes show how closely the two correspond.

In the top FRF, which compares the standard substructuring method to the truth model, one can see that the substructured prediction closely follows the truth data throughout the frequency range shown; up to 2000 [Hz]. The additional peak in the substructured model at 1500 [Hz] is caused by a torsional mode of the cantilever beam; this was not observable in the truth data due to the placement of sensors on the beam. The MAC plot generally agrees with this, as there is strong correlation for most of the truth modes. The seemingly duplicated truth modes, such as 4, 5, and 6, and 7 and 8, are the result of internal shaker modes producing identical motion when viewed from the measurement points on the half cube. The substructure modes under 100 [Hz] do not correspond to any truth modes in the MAC plot due to these not having been significant enough to curve fit in the measured truth data.

When moving down to the middle row of plots, which show the 'MATLAB' results where the decoupling and coupling steps are separated, it can be seen that the resultant model is essentially the same as the 'Control' model. The only significant difference is observed at low frequency, where the 'MATLAB' result has increased the frequency of the modes. This is likely a consequence of the correction that was applied to the stiffness matrix of the decoupled model; a small amount of stiffness was added to correct for negative eigenvalues, which in turn increased the frequency of the modes.

Comparing these two to the bottom row of plots, which present the model from Abaqus, there is good agreement under 800 [Hz], with the same increase in frequency with the modes under 100 [Hz] that was seen in the 'MATLAB' model. Between 800 [Hz] and 1500 [Hz], the Abaqus result seems to be shifted to a lower frequency, or simply not well predicted at all. From 1500 [Hz] to 1800 [Hz], the Abaqus model seems to have generated a set of modes that are not present in either of the other models, or in the truth data. However, the Abaqus result does then correctly predict the large peak near 1900 [Hz]. The MAC plot agrees with these results, in that the modes between 100 [Hz] and 800 [Hz] are correct relative to the truth model, and while those between 800 [Hz] and 1800 [Hz] vaguely resemble the trend shown by the other models, there is worse correlation to the truth model.

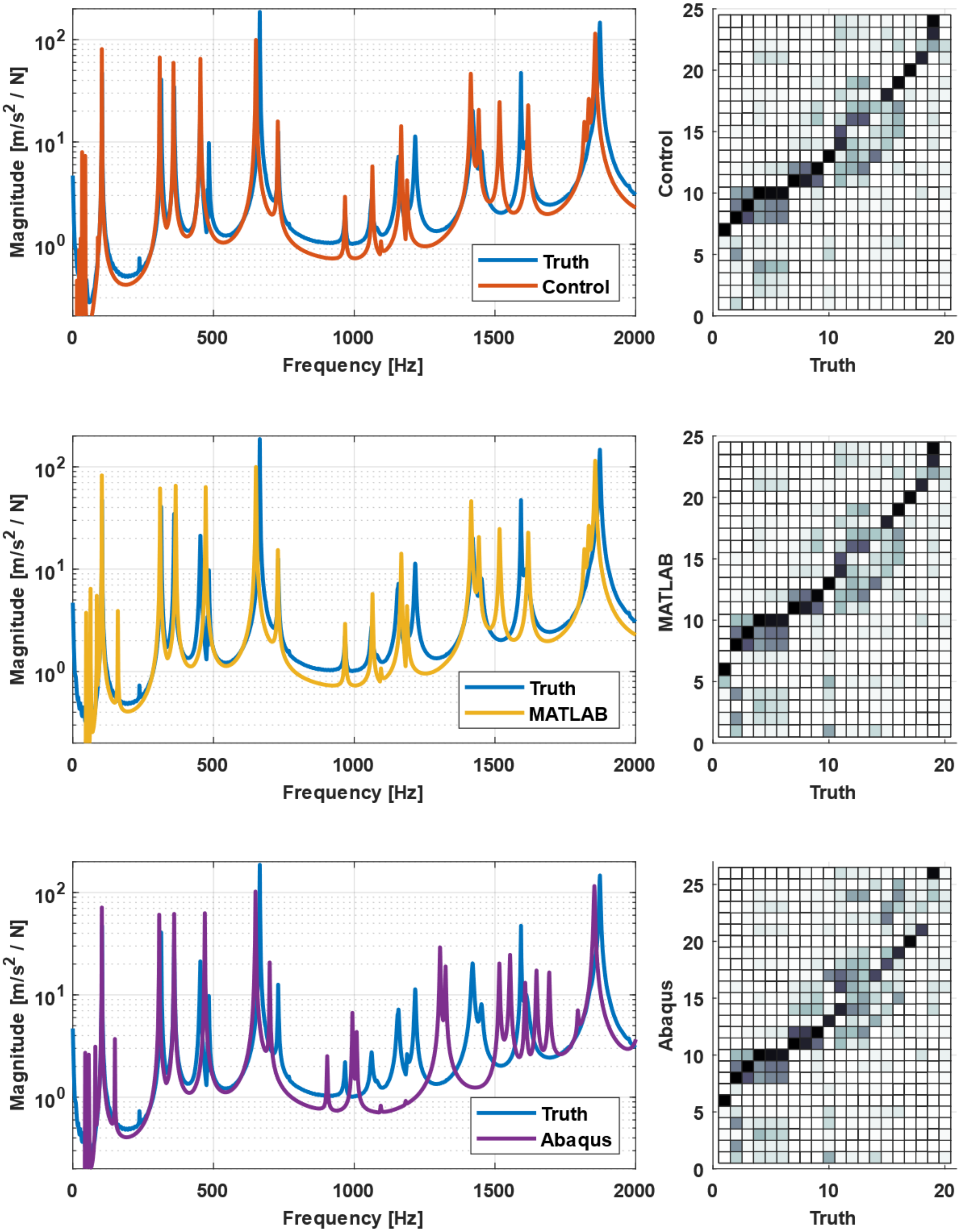

Figure 10: Comparison of half cube substructuring methods; reconstructed FRFs and MACs

As an additional, more focused, evaluation of the substructuring methods, each was used to generate a drive point FRF at the tip of the cantilever beam in its primary bending direction. As before, a damping ratio of 0.1% was assumed while computing the FRF. The results from the truth test and all three substructured models are given below in Figure 11. Unlike the results previously discussed above, the beam tip FRFs from all three substructuring models agree very well with each other and to the truth data, up through 2500 [Hz]; the component of interest that is being substructured to the experimental subsystem, the cantilever, is accurately modeled by Abaqus.

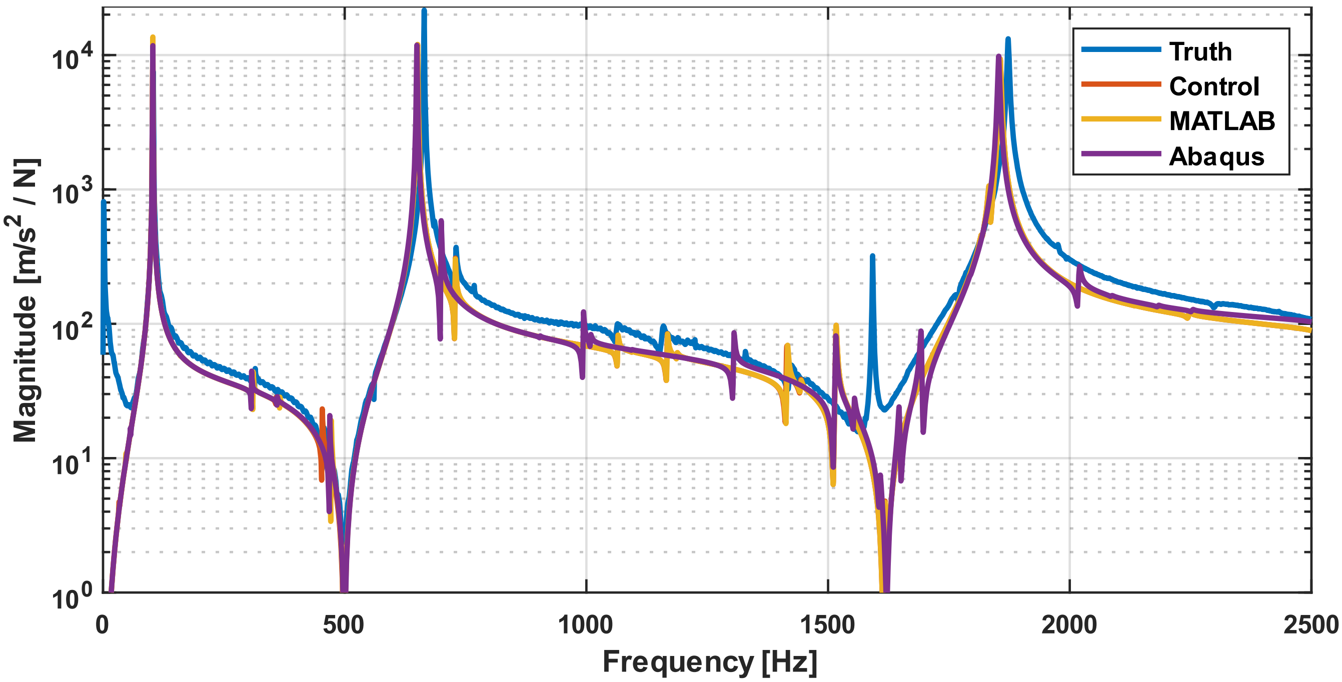

Figure 11: Reconstructed FRFs at the beam tip

The key strength of performing the substructuring computations in Abaqus is the ability to utilize its extensive capabilities for post-processing FEM results. In Abaqus, one can easily visualize the resultant substructured model in the form of the entire analytical subsystem finite element mesh, whereas attempting to do the same in MATLAB would require complex custom scripts and be much more computationally expensive. As a demonstration of the how these results can be viewed in Abaqus, Figure 12 shows an image from Abaqus/CAE of the 700 [Hz] mode of the substructured model. Instead of inferring how the structure is responding based on reconstructed FRFs or unwieldy plots in MATLAB, one can easily observe the dynamic motion of the model directly via the deformed FEM mesh. For this mode, the free corners of the two half cube side walls are flexing while the base remains nearly fixed, as it is now modeling a bolted connection to the shaker armature. Also, since the second cantilever beam mode immediately precedes this at 650 [Hz], the beam is still approximately responding in that shape. Gleaning the same information from the MATLAB model would likely be significantly more cumbersome.

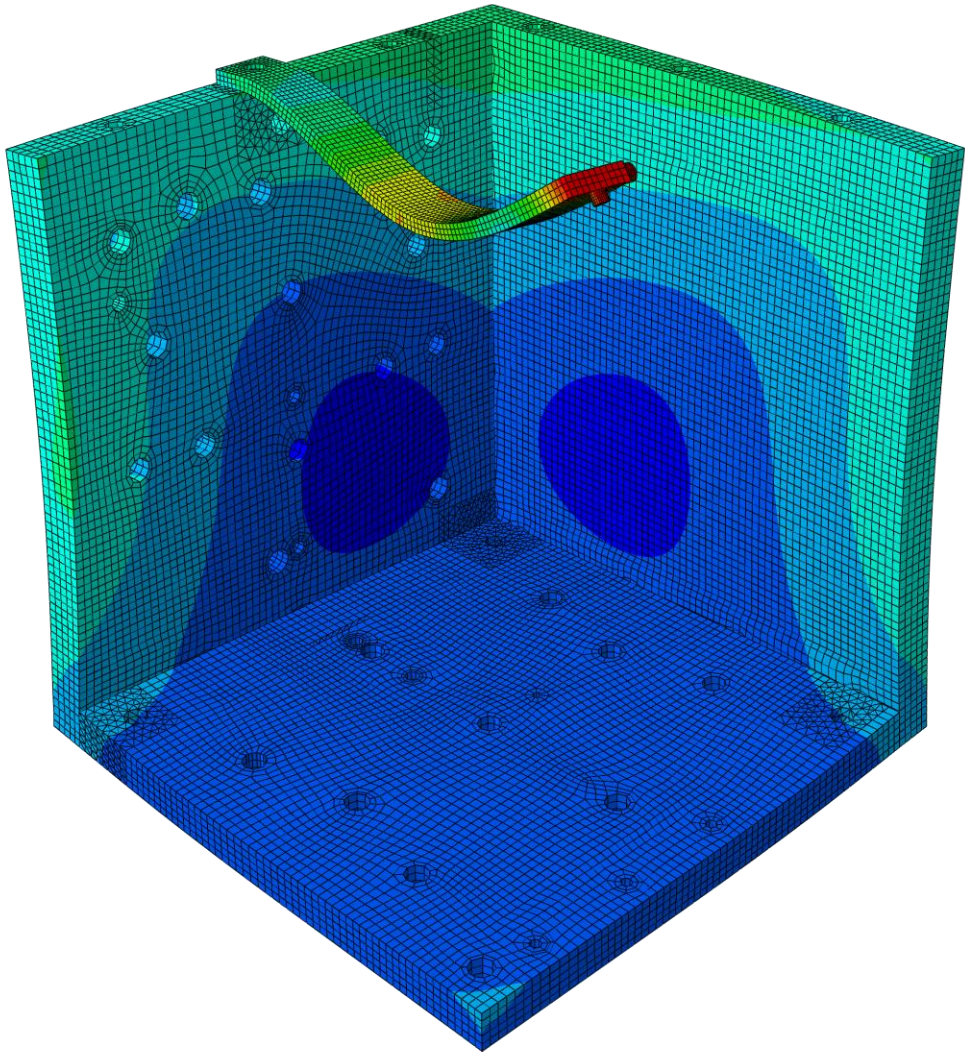

Figure 12: Substructuring result at 700 Hz visualized in Abaqus/CAE

This experimental test case presents a difficult challenge; the TS and experimental subsystem are both quite dynamically complex. When combined in MATLAB in the standard way, the resulting model agrees quite well with the associated truth data. The same can be said about the results when the substructuring computations are separated into two distinct steps in MATLAB. However, when the decoupled model is imported into Abaqus and assembled to the analytical subsystem there, the accuracy of the results for the half cube seem to decrease, while at the same time, the dynamics of the cantilever beam are perfectly represented. The degraded results may, in some way, be due to the analytical subsystem FEM in Abaqus implicitly containing every mode of the model. To assemble the decoupled model and the analytical subsystem in MATLAB, only the first 100 modes of the latter are exported from Abaqus to represent it. If the mathematical computations to assemble the components are effectively the same whether done in MATLAB or Abaqus, as was the situation in the numerical case study, the only difference between the two substructuring methods is how many modes are contained in the analytical subsystem. This conclusion seems contrary to the generally accepted substructuring guideline that the resultant model should become more accurate as modes are added to the analytical subsystem.

CONCLUSIONS & FUTURE WORK

A standard approach for implementing the TS method of experimental substructuring is to import and assemble all the subsystems in MATLAB. The preceding sections presented a procedure for performing this in Abaqus, producing a model that can leverage the full post-processing capabilities of a FEM. This was accomplished by separating the usually simultaneously computed substructuring constraints into two separate actions; decouple the TS from the experimental subsystem in MATLAB and then couple the result to the analytical subsystem FEM in Abaqus. This approach introduces some intricacies, as it is likely that a correction must be applied to the decoupled model before it can be imported into Abaqus.

When applied to a numerical case study with simple beam models, the proposed method was found to yield excellent results that agreed with those from the standard substructuring method as well as a set of truth data. However, contrary to standard TS method substructuring practices, this was achieved through limiting what modes are included in the TS and experimental subsystem modal bases, such that the correction to the decoupled model was minimized. While this resulted in near perfect agreement between all models, it likely limited their applicable frequency range by restricting the modal bases.

The process was then implemented on a complex experimental test case in which a FEM of a shaker adapter and potential test setup, a half cube with an attached cantilever beam, was attached to an experimental model of the half cube adapter mounted to an electrodynamic shaker. In this case, while the standard and separated approaches in MATLAB agree very well, the results from Abaqus are noticeably less accurate for the half cube, but still very accurate for the cantilever beam. The cause of this discrepancy is possibly due to the Abaqus FEM implicitly containing every mode of the half cube and beam system, as only the first 100 modes were imported into MATLAB for assembly there. This is an additional instance of observed results seemingly contradicting common substructuring practices.

This work shows that the TS method of experimental substructuring can be implemented in Abaqus by importing an experimental model with decoupled TS from MATLAB and constraining it to the native Abaqus FEM. This process is limited by the need to correct the decoupled model before importing it into Abaqus, reducing the TS and experimental subsystem modal bases to minimize this correction, and the possibility that several normal substructuring practices seem to be detrimental to the results. Thus, while some accuracy penalty is likely to be observed, experimental substructuring can be completed within Abaqus, allowing for expanded accessibility and much more practical results. Future work includes establishing metrics for assessing the quality of the decoupled experimental subsystem and exploring other options for importing that model into Abaqus, such as generating it as a Hurty/Craig-Bampton model that may be imported as a standard super-element into Abaqus.

SANDIA ACKNOWLEDGEMENT

Notice: This manuscript has been authored by National Technology and Engineering Solutions of Sandia, LLC. under Contract No. DE-NA0003525 with the U.S. Department of Energy/National Nuclear Security Administration. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

REFERENCES

[1] D. de Klerk and e. al., "General framework for dynamic substructuring: History, review, and classification of techniques," *AIAA Journal,* vol. 46, pp. 1169-1181, 2008.

[2] M. S. Allen, R. L. Mayes and E. J. Bergman, "Experimental Modal Substructuring to Couple and Uncouple Substructures with Flexible Fixtures and Multi-point Connections," *Journal of Sound and Vibration,* vol. 329, p. 4891–4906, 2010.

[3] "Dynamic Substructuring Wiki," [Online]. Available: http://substructure.engr.wisc.edu.

[4] M. S. Allen and R. L. Mayes, "Comparison of FRF and Modal Methods for Combining Experimental and Analytical Substructures," in *Proceedings of the 25th IMAC*, Orlando, Florida, 2007.

[5] R. L. Mayes and M. Arviso, "Design Studies for the Transmission Simulator Method of Experimental Dynamic Substructuring," in *International Seminar on Modal Analysis*, Lueven, Belgium, 2010.

[6] D. S. SIMULIA, "Abaqus Analysis User's Guide," 2016. [Online]. Available: http://130.149.89.49:2080/v2016/books/usb/default.htm. [Accessed 30 Setember 2019].

[7] R. L. Mayes, M. S. Allen and D. C. Kammer, "Correcting Indefinite Mass Matrices Due to Substructure Uncoupling," *Journal of Sound and Vibration,* pp. 5856-5866, 2013.

[8] M. S. Allen, D. C. Kammer and R. L. Mayes, "Metrics for diagnosing negative mass and stiffness when uncoupling experimental and analytical substructures," *Journal of Sound and Vibration,* vol. 331, p. 5435–5448, 2012.

[9] B. Moldenhauer, M. S. Allen, W. J. DeLima and E. Dodgen, "Using Hybrid Modal Substructuring with a Complex Transmission Simulator to Model an Electrodynamic Shaker," in *Proceedings of the 37th IMAC*, Orlando, Florida, 2019.

[10] D. C. Kammer, "Sensor placement for on-orbit modal identification and correlation of large space structures," *Journal of Guidance, Control, and Dynamics,* vol. 14, no. 2, pp. 251-259, 1991.

APPENDIX

APPENDIX A: MATLAB FUNCTION TO GENERATE AUXILLARY ABAQUS INPUT FILE

```
function ABQ_Constr(filename,constr,fn,FEM_DOF,Num_FEM_Nodes,Num_FEM_Eles)
%   Writes a text file for performing experimental substructuring in Abaqus.
% Generates nodes, constrains DOF 2-6, assigns unit mass, attaches a
% grounded spring with stiffness equal to the square of the decoupled model
% natural frequencies, then writes out constraint equations as Linear
% Equation Multipoint Constraints.
%
% INPUTS
%   filename - name of written text file
%   constr - Constraint Equations in RREF
%   fn - Natural Frequencies of decoupled model [Hz]
%   FEM_DOF - DOF in Abaqus FEM to constrain to
%   Num_FEM_Nodes - Number of nodes in the Abaqus FEM
%   Num_FEM_Eles - Number of elements in the Abaqus FEM
%
% OUTPUTS
%   Writes a text file to the current directory
%
% In the .inp file for the Abaqus FEM, include the following line:
% *INCLUDE,INPUT=filename.inp
% Both .inp files must be in the same directory.
%
% EXAMPLE OF PROPER INPUTS
% filename = 'example.inp';
% constr = [1 0 .5 .5 ; 0 1 .5 .5];
% fn = [1 2];
% FEM_DOF = [1.2 2.2]; % 2nd DOF in nodes 1 and 2
% Num_FEM_Nodes = 2;
% Num_FEM_Eles  = 1;


    % Open Text file to write Abaqus Auxiliary Input File
fid = fopen(filename,'wt');


    % Create the Modal Nodes
% Number of Modal Nodes
Num_Q_DOF = size(fn(:),1);
% Modal Nodes
Q_nodes = [((Num_FEM_Nodes+1):(Num_FEM_Nodes+Num_Q_DOF)).',zeros(Num_Q_DOF,3)];
% Reset local coord system to global
fprintf(fid,'*SYSTEM\n');
% Initialize node creation
fprintf(fid,'*NODE\n');
% Write nodes
fprintf(fid,'%i,\t%.8g,\t%.8g,\t%.8g\n',Q_nodes');


    % Assign Boundary Conditions
% Initialize Node Set
fprintf(fid,'*NSET, NSET=Modal_DOF, GENERATE\n');
% Create Node Set
fprintf(fid,'%i, %i, 1\n',Q_nodes(1,1),Q_nodes(end,1));
% Initialize Boundary definitions
```

```
fprintf(fid,'*BOUNDARY\n');
% Constrain DOF 2-6, DOF 1 will be used for spring stiffness
fprintf(fid,'Modal_DOF, 2, 6\n');


    % Assign Point Mass
% Modal Mass Elements
Q_m_eles = [((Num_FEM_Eles+1):(Num_FEM_Eles+Num_Q_DOF)).' Q_nodes(1:end,1)];
% Initialize Point Mass Element Set
fprintf(fid,'*Element, type=MASS, elset=Modal_Masses\n');
% Write Mass Elements
fprintf(fid,'%i,\t%.8g\n',Q_m_eles');
% Initialize Point Mass Element Set
fprintf(fid,'*Mass, elset=Modal_Masses\n1,\n');


    % Assign Grounded Spring
% Modal Stiffness Elements
Q_k_eles = [((Q_m_eles(end,1)+1):(Q_m_eles(end,1)+Num_Q_DOF)).',Q_nodes(1:end,1)];
for ii=1:Num_Q_DOF
    fprintf(fid,'*Spring, elset=K%i\n',ii); % Initialize Spring Element
    fprintf(fid,'1\n'); % Spring DOF
    fprintf(fid,'%.19G\n',(fn(ii)*2*pi)^2); % Spring Stiffness (Maximum 20
characters for a data line)
    fprintf(fid,'*Element, type=Spring1, elset=K%i\n',ii); % Grounded Type Spring
    fprintf(fid,'%i, %i\n',Q_k_eles(ii,1),Q_k_eles(ii,2)); % Spring Element and
Node
end


    % Write Linear Equation Constraints
% DOF to constrain together
constr_DOF = [Q_nodes(1:end,1)+.1 ; FEM_DOF(:)];
% Seperating into Node and Direction
constr_inds = [floor(constr_DOF) round(10*mod(constr_DOF,1))];
for jj=1:size(constr,1)
    % Indices of Nonzero Equation Terms
    term_ind = find(constr(jj,:));
    % Initialize Equation Field
    fprintf(fid,'*EQUATION\n');
    % Number of Terms in the Equation
    fprintf(fid,'%i\n',length(term_ind));
    % Write Equation Terms
    fprintf(fid,'%i,\t%i,\t%.19G,\n',[constr_inds(term_ind(1:end),1),...
        constr_inds(term_ind(1:end),2),constr(jj,term_ind(1:end))']');
end


    % Close text file
fclose(fid);

end
```

APPENDIX B: TRUNCATED AUXILLARY ABAQUS INPUT FILE FOR BEAM CASE STUDY

```
*SYSTEM
*NODE
5001,   0,  0,  0

…

5014,   0,  0,  0
*NSET, NSET=Modal_DOF, GENERATE
5001, 5014, 1
*BOUNDARY
Modal_DOF, 2, 6
*Element, type=MASS, elset=Modal_Masses
5001,   5001

…

5014,   5014
*Mass, elset=Modal_Masses
1,
*Spring, elset=K1
1
242212591293406.625
*Element, type=Spring1, elset=K1
5015, 5001

…

*Spring, elset=K14
1
99889894.06944584846
*Element, type=Spring1, elset=K14
5028, 5014
*EQUATION
11
5001,   1,  1,
5006,   1,  -0.000264860351395888741,
5007,   1,  -0.000102541765432669971 4,
5008,   1,  -0.000157111482203310410 4,
5009,   1,  -0.000208173121393683219 3,
5010,   1,  0.000287025022707926953 1,
5011,   1,  0.000333786756687496223 1,
5012,   1,  -0.000382954404601637261 7,
1,  2,  -0.000127177405682847523 2,
51, 2,  4.163977166513179649E-05,
101,    2,  -8.145943855598651294E-06,

…

*EQUATION
6
5005,   1,  1,
5013,   1,  -0.9985942660576510033,
5014,   1,  0.9774124704222225679,
1,  1,  -0.389357596507627024,
51, 1,  -0.106769751316806466,
101,    1,  0.1758183602094033071,
```