

An Abaqus-Matlab Tutorial for Jointed Systems

International Committee on Jointed Structures Seminar Series

Nidish Narayanaa Balaji

Department of Aerospace Engineering, IIT Madras

March 26, 2025

Table of Contents

- 1 Introduction
- 2 Outline of the Steps
 - Relative Coordinates Pipeline
- 3 Nonlinear Analysis in MATLAB
- 4 Outro and Recommendations



*Detailed Instructions are hosted through Github
in a repository named [Nidish96/Abaqus4Joints](#)*

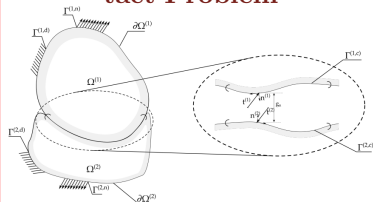
Acknowledgements

- Prof. Matthew Brake
- Dr. Justin Porter, Maeve Karpov
- Prof. Matt Allen

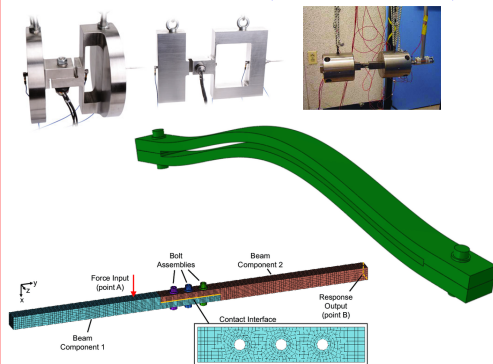
The research needs of the joints community are quite unique and specific

1. Introduction

The Generic Contact Problem



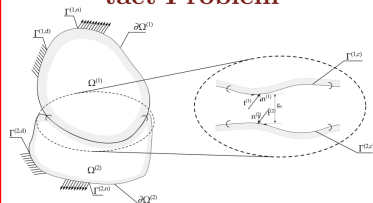
Joints Benchmarks (see jointmechanics.org)



The research needs of the joints community are quite unique and specific

1. Introduction

The Generic Contact Problem

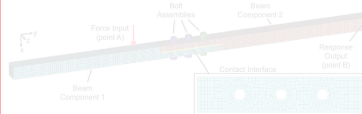


Joints Benchmarks (see jointmechanics.org)



Common theme:

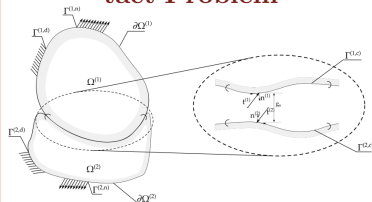
Linear subcomponents joined through a non-linear contact interface undergoing small deformation vibrations



The research needs of the joints community are quite unique and specific

1. Introduction

The Generic Contact Problem



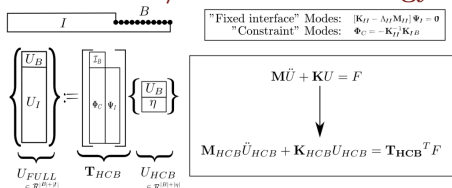
Joints Benchmarks (see jointmechanics.org)



Common theme:

Linear subcomponents joined through a non-linear contact interface undergoing small deformation vibrations

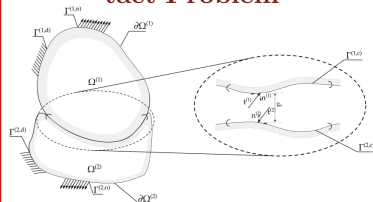
The HCB/CMS Methodology



The research needs of the joints community are quite unique and specific

1. Introduction

The Generic Contact Problem



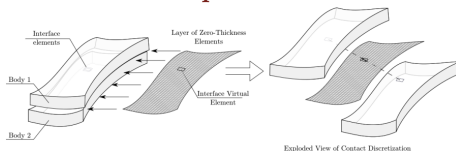
Joints Benchmarks (see jointmechanics.org)



Common theme:

Linear subcomponents joined through a non-linear contact interface undergoing **small deformation vibrations**

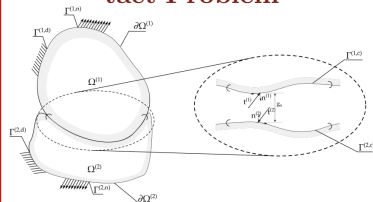
Interface Virtual Element Representations



The research needs of the joints community are quite unique and specific

1. Introduction

The Generic Contact Problem

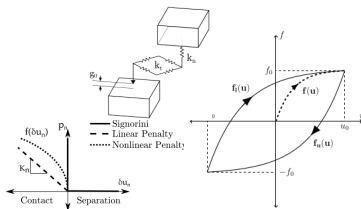


Joints Benchmarks (see jointmechanics.org)

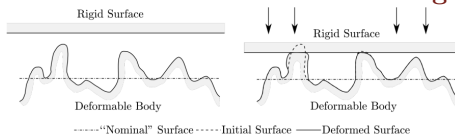


Common theme:

Linear subcomponents joined through a **non-linear contact interface** undergoing small deformation vibrations



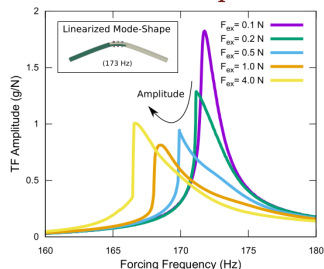
Interface Constitutive Modeling



The research needs of the joints community are quite unique and specific

1. Introduction

Nonlinear Responses

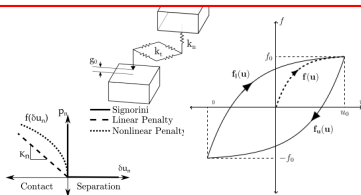
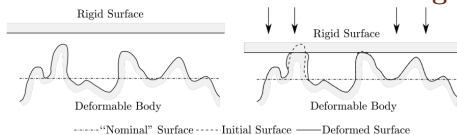


Joints Benchmarks (see jointmechanics.org)



Linear subcomponents joined through a non-linear contact interface undergoing small deformation vibrations

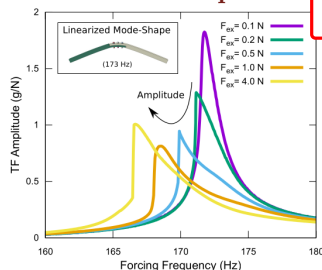
Interface Constitutive Modeling



The research needs of the joints community are quite unique and specific

1. Introduction

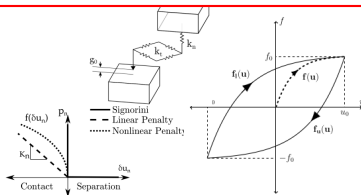
Nonlinear Responses



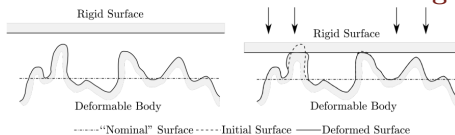
Goal: Off-Load as much as possible to **focus on joints**.

Common theme:

Linear subcomponents joined through a **non-linear contact interface** undergoing small deformation vibrations



Interface Constitutive Modeling

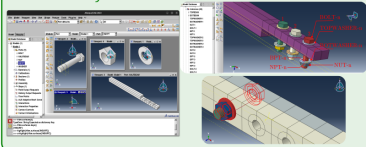


An Overview of the Modeling Pipeline

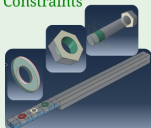
2. Outline of the Steps

Pre-Processing

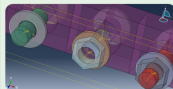
Assembly



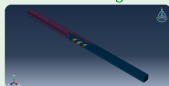
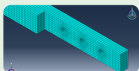
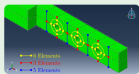
Constraints



Bolt Prestress as Static Load



Seeded Meshing, Processing



Component Mode Synthesis



$$\begin{aligned} \text{"Fixed interface" Modes: } & [K_{FI} - \lambda_0 M_{FI}] \Phi_I = 0 \\ \text{"Constraint" Modes: } & \Phi_C = -K_{CI}^{-1} K_{IB} \end{aligned}$$

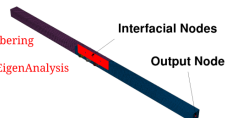
$$\begin{Bmatrix} U_B \\ U_I \end{Bmatrix} = \begin{Bmatrix} I_n \\ \Phi_I \end{Bmatrix} \begin{Bmatrix} U_B \\ \eta \end{Bmatrix}$$

$$\underbrace{U_B}_{U_{FULL} \in \mathbb{R}^{N \times 1}} \quad \underbrace{\Phi_I}_{T_{HCB}} \quad \underbrace{U_{HCB}}_{U_{HCB} \in \mathbb{R}^{N_{HCB} \times 1}}$$

$$M \ddot{U} + KU = F$$

$$\downarrow$$

$$M_{HCB} \ddot{U}_{HCB} + K_{HCB} U_{HCB} = T_{HCB}^T F$$



- + Reorganize NodeSet numbering
- + Choose recovery nodes
- + Conduct Fixed Interface EigenAnalysis

Matrix Extraction

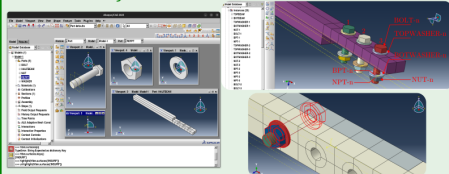
- + Use the provided python script to convert the mtx files into MATLAB mat files

```
1: #
2: #
3: # STEP: HCB000
4: #
5: # Create model from HCB000
6: # Substructure generation, using the HCB000, recovery matrices,
7: # and HCB000, and HCB000
8: # Create Eigenmodes, generate
9: # HCB000, generate, HCB000, HCB000, HCB000
10: # HCB000, HCB000, HCB000
11: # HCB000, HCB000, HCB000
12: # HCB000, HCB000, HCB000
13: # HCB000, HCB000, HCB000
14: # HCB000, HCB000, HCB000
15: # HCB000, HCB000, HCB000
16: # HCB000, HCB000, HCB000
17: # HCB000, HCB000, HCB000
18: # HCB000, HCB000, HCB000
19: # HCB000, HCB000, HCB000
20: # HCB000, HCB000, HCB000
21: # HCB000, HCB000, HCB000
22: # HCB000, HCB000, HCB000
23: # HCB000, HCB000, HCB000
24: # HCB000, HCB000, HCB000
25: # HCB000, HCB000, HCB000
26: # HCB000, HCB000, HCB000
27: # HCB000, HCB000, HCB000
28: # HCB000, HCB000, HCB000
29: # HCB000, HCB000, HCB000
30: # HCB000, HCB000, HCB000
31: # HCB000, HCB000, HCB000
32: # HCB000, HCB000, HCB000
33: # HCB000, HCB000, HCB000
34: # HCB000, HCB000, HCB000
35: # HCB000, HCB000, HCB000
36: # HCB000, HCB000, HCB000
37: # HCB000, HCB000, HCB000
38: # HCB000, HCB000, HCB000
39: # HCB000, HCB000, HCB000
40: # HCB000, HCB000, HCB000
41: # HCB000, HCB000, HCB000
42: # HCB000, HCB000, HCB000
43: # HCB000, HCB000, HCB000
44: # HCB000, HCB000, HCB000
45: # HCB000, HCB000, HCB000
46: # HCB000, HCB000, HCB000
47: # HCB000, HCB000, HCB000
48: # HCB000, HCB000, HCB000
49: # HCB000, HCB000, HCB000
50: # HCB000, HCB000, HCB000
51: # HCB000, HCB000, HCB000
52: # HCB000, HCB000, HCB000
53: # HCB000, HCB000, HCB000
54: # HCB000, HCB000, HCB000
55: # HCB000, HCB000, HCB000
56: # HCB000, HCB000, HCB000
57: # HCB000, HCB000, HCB000
58: # HCB000, HCB000, HCB000
59: # HCB000, HCB000, HCB000
60: # HCB000, HCB000, HCB000
61: # HCB000, HCB000, HCB000
62: # HCB000, HCB000, HCB000
63: # HCB000, HCB000, HCB000
64: # HCB000, HCB000, HCB000
65: # HCB000, HCB000, HCB000
66: # HCB000, HCB000, HCB000
67: # HCB000, HCB000, HCB000
68: # HCB000, HCB000, HCB000
69: # HCB000, HCB000, HCB000
70: # HCB000, HCB000, HCB000
71: # HCB000, HCB000, HCB000
72: # HCB000, HCB000, HCB000
73: # HCB000, HCB000, HCB000
74: # HCB000, HCB000, HCB000
75: # HCB000, HCB000, HCB000
76: # HCB000, HCB000, HCB000
77: # HCB000, HCB000, HCB000
78: # HCB000, HCB000, HCB000
79: # HCB000, HCB000, HCB000
80: # HCB000, HCB000, HCB000
81: # HCB000, HCB000, HCB000
82: # HCB000, HCB000, HCB000
83: # HCB000, HCB000, HCB000
84: # HCB000, HCB000, HCB000
85: # HCB000, HCB000, HCB000
86: # HCB000, HCB000, HCB000
87: # HCB000, HCB000, HCB000
88: # HCB000, HCB000, HCB000
89: # HCB000, HCB000, HCB000
90: # HCB000, HCB000, HCB000
91: # HCB000, HCB000, HCB000
92: # HCB000, HCB000, HCB000
93: # HCB000, HCB000, HCB000
94: # HCB000, HCB000, HCB000
95: # HCB000, HCB000, HCB000
96: # HCB000, HCB000, HCB000
97: # HCB000, HCB000, HCB000
98: # HCB000, HCB000, HCB000
99: # HCB000, HCB000, HCB000
100: # HCB000, HCB000, HCB000
```

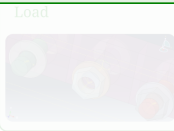
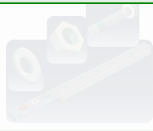
An Overview of the Modeling Pipeline

2. Outline of the Steps

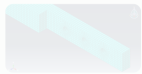
Pre-Processing Assembly



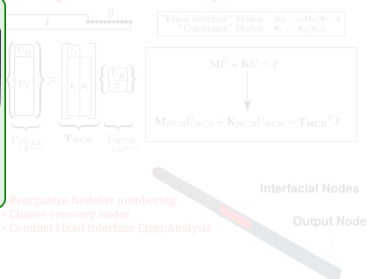
Load



Seeded Meshing. Processing



Component Mode Synthesis



Matrix Extraction

+ Use the provided python script to convert the mtx files into MATLAB mat files

```

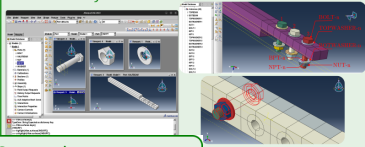
1 #!/usr/bin/env python
2 """
3 This script converts the Abaqus .mtx files into MATLAB .mat files.
4 It uses the provided python script to convert the mtx files into MATLAB mat files.
5 """
6 import sys
7 import os
8 import glob
9 import numpy as np
10 import scipy.io as io
11
12 # Get the directory where this script is located
13 script_dir = os.path.dirname(os.path.abspath(__file__))
14
15 # Get the list of .mtx files in the directory
16 mtx_files = glob.glob(os.path.join(script_dir, "*.mtx"))
17
18 # Loop through the .mtx files and convert them to .mat files
19 for mtx_file in mtx_files:
20     # Extract the base name of the .mtx file
21     base_name = os.path.splitext(os.path.basename(mtx_file))[0]
22
23     # Create the .mat file name
24     mat_file = base_name + ".mat"
25
26     # Read the .mtx file
27     mtx = np.loadtxt(mtx_file)
28
29     # Write the .mat file
30     io.savemat(mat_file, {'mtx': mtx})
31
32     print(f"Converted {mtx_file} to {mat_file}")
33
34 # End of script
  
```

An Overview of the Modeling Pipeline

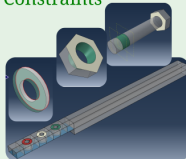
2. Outline of the Steps

Pre-Processing

Assembly



Constraints

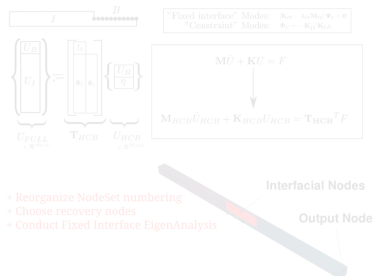


Bolt Prestress as Static Load

Refined Meshing, Processing



Component Mode Synthesis



Matrix Extraction

+ Use the provided python script to convert the mtx files into MATLAB mat files

```

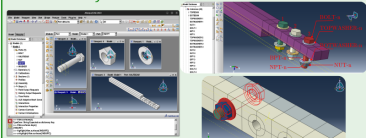
1 #!/usr/bin/env python
2 """
3 This script extracts the matrices from the Abaqus .mtx files
4 and converts them into MATLAB .mat files.
5 """
6 import sys
7 import os
8 import glob
9 import numpy as np
10
11 # Define the directory where the .mtx files are located
12 mtz_dir = 'mtz_files'
13
14 # Define the directory where the .mat files will be saved
15 mat_dir = 'mat_files'
16
17 # Create the mat_dir directory if it doesn't exist
18 if not os.path.exists(mat_dir):
19     os.makedirs(mat_dir)
20
21 # Loop through all .mtx files in the mtz_dir
22 for mtz_file in glob.glob(mtz_dir + '/*.mtx'):
23     # Extract the matrix name from the file name
24     matrix_name = os.path.splitext(os.path.basename(mtz_file))[0]
25
26     # Read the matrix from the .mtx file
27     matrix = np.loadtxt(mtz_file)
28
29     # Save the matrix to a .mat file
30     np.save(os.path.join(mat_dir, matrix_name + '.mat'), matrix)
31
32 # Print the number of matrices extracted
33 print('Number of matrices extracted: ' + str(len(glob.glob(mtz_dir + '/*.mtx'))))
34
35 """

```

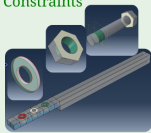
An Overview of the Modeling Pipeline

2. Outline of the Steps

Pre-Processing



Constraints



Bolt Prestress as Static Load



- + Reorganize NodeSet numbering
- + Choose recovery nodes
- + Com...

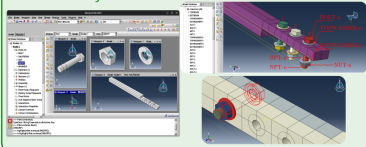
"Pull the Bolts, Push the Washers"

An Overview of the Modeling Pipeline

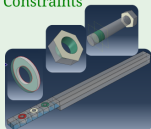
2. Outline of the Steps

Pre-Processing

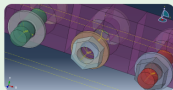
Assembly



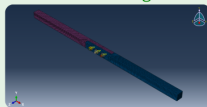
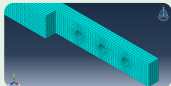
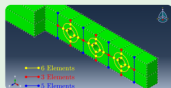
Constraints



Bolt Prestress as Static Load



Seeded Meshing, Processing

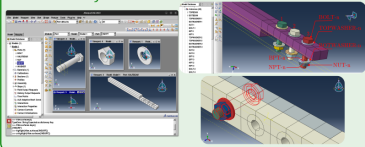


An Overview of the Modeling Pipeline

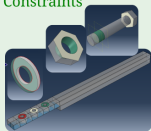
2. Outline of the Steps

Pre-Processing

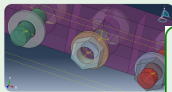
Assembly



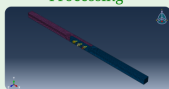
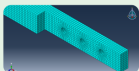
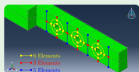
Constraints



Bolt Prestress as Static Load



Seeded Meshing, Processing



Component Mode Synthesis



"Fixed interface" Modes: $[K_{II} - \lambda_p M_{II}] \Phi_I = 0$
 "Constraint" Modes: $\Phi_C = -K_{IC}^{-1} K_{IB}$

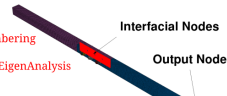
$$\begin{Bmatrix} U_B \\ U_I \end{Bmatrix} = \begin{Bmatrix} I_n \\ \Phi_I \end{Bmatrix} \begin{Bmatrix} U_B \\ \eta \end{Bmatrix}$$

$$\underbrace{U_B}_{U_{FULL} \in \mathbb{R}^{R \times F}} \quad \underbrace{\Phi_I}_{T_{HCB}} \quad \underbrace{U_{HCB}}_{U_{HCB} \in \mathbb{R}^{R \times H}}$$

$$M \ddot{U} + KU = F$$

$$\downarrow$$

$$M_{HCB} \ddot{U}_{HCB} + K_{HCB} U_{HCB} = T_{HCB}^T F$$



- + Reorganize NodeSet numbering
- + Choose recovery nodes
- + Conduct Fixed Interface EigenAnalysis

Matrix Extraction

- + Use the provided python script to convert the mtx files into MATLAB mat files

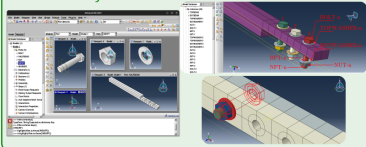
```
1: **
2: **
3: ** STEP: HCBCHS
4: **
5: *vplot, name=HCBCHS, plot=HCBCHS
6: *Substructure Generate, name=HCBCHS, type=2, recovery matrix=YES,
7:   *Fixed-Interface Modes, name=HCBCHS
8: *Select Eigenmodes, generate
9: 1, 20, 1
10: *Damping Controls, structural=CONROD, viscous=CONROD
11: *Material Model Data, sorted=0
12: BOTS, MOD, 1, 3
13: TOTS, MOD, 1, 3
14: **
15: ** LOAD CASES
16: **
17: *Substructure Load Case, name=LCASE
18: **
19: *vplot
20: BPT-1 Set-1, 1, 1,
21: ** Name: BoltLoad-1 Type: Concentrated force Scale factor: 1
22: *vplot
23: BPT-2 Set-1, 1, 1,
24: ** Name: BoltLoad-2 Type: Concentrated force Scale factor: 1
25: *vplot
26: BPT-3 Set-1, 1, 1,
27: ** Name: BoltLoad-3 Type: Concentrated force Scale factor: 1
28: *vplot
29: BPT-1 Set-1, 1, -1,
30: ** Name: BoltLoad-1 Type: Concentrated force Scale factor: 1
31: *vplot
32: BPT-2 Set-1, 1, -1,
33: ** Name: BoltLoad-2 Type: Concentrated force Scale factor: 1
34: *vplot
35: BPT-3 Set-1, 1, -1,
36: ** Name: BoltLoad-3 Type: Concentrated force Scale factor: 1
37: *Substructure Matrix Output, FILE NAME=HCBCHS, HCBCHS=YES,
38:   *STIFFNESS=YES, SLASH=YES, RECOVERY MATRIX=YES
39: ** End Step
```

An Overview of the Modeling Pipeline

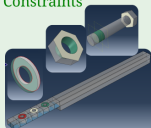
2. Outline of the Steps

Pre-Processing

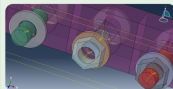
Assembly



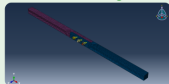
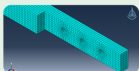
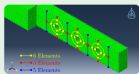
Constraints



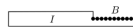
Bolt Prestress as Static Load



Seeded Meshing, Processing



Component Mode Synthesis



$$\begin{aligned} \text{"Fixed interface" Modes: } & [K_{FI} - \lambda_0 M_{FI}] \Psi_I = 0 \\ \text{"Constraint" Modes: } & \Phi_C = -K_{CI}^{-1} K_{IB} \end{aligned}$$

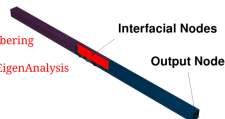
$$\begin{Bmatrix} U_B \\ U_I \end{Bmatrix} = \begin{Bmatrix} I_n \\ \Phi_C \end{Bmatrix} \begin{Bmatrix} U_B \\ \eta \end{Bmatrix}$$

$$\underbrace{U_B}_{U_{FULL} \in \mathbb{R}^{N \times 1}} \quad \underbrace{\Phi_C}_{T_{HCB}} \quad \underbrace{U_B}_{U_{HCB} \in \mathbb{R}^{N \times 1}}$$

$$M \ddot{U} + KU = F$$

$$\downarrow$$

$$M_{HCB} \ddot{U}_{HCB} + K_{HCB} U_{HCB} = T_{HCB}^T F$$



- + Reorganize NodeSet numbering
- + Choose recovery nodes
- + Conduct Fixed Interface EigenAnalysis

Matrix Extraction

- + Use the provided python script to convert the mtx files into MATLAB mat files

```
1: #
2: #
3: # STEP: WCB000
4: #
5: # Step: WCB000, eigenmode
6: # Substructure generation, using the type=2, recovery matrices.
7: # Use the following command:
8: #
9: # *Eigen, generate
10: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
11: # *Eigen, mode, 1, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
12: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
13: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
14: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
15: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
16: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
17: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
18: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
19: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
20: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
21: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
22: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
23: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
24: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
25: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
26: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
27: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
28: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
29: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
30: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
31: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
32: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
33: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
34: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
35: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
36: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
37: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
38: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
39: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
40: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
41: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
42: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
43: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
44: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
45: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
46: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
47: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
48: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
49: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
50: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
51: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
52: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
53: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
54: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
55: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
56: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
57: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
58: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
59: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
60: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
61: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
62: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
63: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
64: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
65: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
66: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
67: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
68: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
69: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
70: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
71: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
72: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
73: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
74: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
75: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
76: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
77: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
78: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
79: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
80: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
81: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
82: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
83: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
84: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
85: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
86: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
87: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
88: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
89: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
90: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
91: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
92: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
93: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
94: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
95: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
96: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
97: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
98: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
99: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
100: # *E, 20, 5, generate, eigenmode=2, eigenmode=2, eigenmode=2
```


2.1. Relative Coordinates Pipeline

Outline of the Steps

- Suppose that $\underline{u}_T, \underline{u}_B, \underline{u}_I$ are the vectors of top, bottom, and internal nodal DoFs.
- Then the governing equations look like:

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ \underline{\underline{M}}_{BT} & \underline{\underline{M}}_{BB} & \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \ddot{\underline{u}}_T \\ \ddot{\underline{u}}_B \\ \ddot{\underline{u}}_I \end{bmatrix} + \begin{bmatrix} \underline{\underline{K}}_{TT} & \underline{\underline{K}}_{TB} & \underline{\underline{K}}_{TI} \\ \underline{\underline{K}}_{BT} & \underline{\underline{K}}_{BB} & \underline{\underline{K}}_{BI} \\ \text{sym} & & \underline{\underline{K}}_{II} \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(c)} \\ \underline{F}_B^{(c)} \\ \underline{0} \end{bmatrix} = \underline{F}_e(t)$$

2.1. Relative Coordinates Pipeline

Outline of the Steps

- Suppose that $\underline{u}_T, \underline{u}_B, \underline{u}_I$ are the vectors of top, bottom, and internal nodal DoFs.
- Then the governing equations look like:

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ \underline{\underline{M}}_{BT} & \underline{\underline{M}}_{BB} & \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \ddot{\underline{u}}_T \\ \ddot{\underline{u}}_B \\ \ddot{\underline{u}}_I \end{bmatrix} + \begin{bmatrix} \underline{\underline{K}}_{TT} & \underline{\underline{K}}_{TB} & \underline{\underline{K}}_{TI} \\ \text{sym} & \underline{\underline{K}}_{BB} & \underline{\underline{K}}_{BI} \\ & & \underline{\underline{K}}_{II} \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(c)} \\ \underline{F}_B^{(c)} \\ \underline{0} \end{bmatrix} = \underline{F}_e(t)$$

- Under the relative coordinate transformation,

$$\begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} = \begin{bmatrix} \underline{\underline{I}}_T & \underline{\underline{I}}_T & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{I}}_T & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} & \underline{\underline{I}}_I \end{bmatrix} \begin{bmatrix} \Delta \underline{u} \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix}, \quad \Delta \underline{u} = \underline{u}_T - \underline{u}_B.$$

2.1. Relative Coordinates Pipeline

Outline of the Steps

- Suppose that $\underline{u}_T, \underline{u}_B, \underline{u}_I$ are the vectors of top, bottom, and internal nodal DoFs.
- Then the governing equations look like:

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ \underline{\underline{M}}_{BT} & \underline{\underline{M}}_{BB} & \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \ddot{\underline{u}}_T \\ \ddot{\underline{u}}_B \\ \ddot{\underline{u}}_I \end{bmatrix} + \begin{bmatrix} \underline{\underline{K}}_{TT} & \underline{\underline{K}}_{TB} & \underline{\underline{K}}_{TI} \\ \underline{\underline{K}}_{BT} & \underline{\underline{K}}_{BB} & \underline{\underline{K}}_{BI} \\ \text{sym} & & \underline{\underline{K}}_{II} \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(e)} \\ \underline{F}_B^{(e)} \\ \underline{0} \end{bmatrix} = \underline{F}_e(t)$$

- Under the relative coordinate transformation,

Transformed Equations of Motion

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TT} + \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ & \underline{\underline{M}}_{TT} + \underline{\underline{M}}_{BB} + \underline{\underline{M}}_{TB} + \underline{\underline{M}}_{BT} & \underline{\underline{M}}_{TI} + \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \ddot{\underline{u}}_T \\ \ddot{\underline{u}}_B \\ \ddot{\underline{u}}_I \end{bmatrix} + \begin{bmatrix} & & \\ & \bar{\underline{K}} & \\ & & \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(e)} \\ \underline{0} \\ \underline{0} \end{bmatrix} = \underline{F}_e(t).$$

2.1. Relative Coordinates Pipeline

Outline of the Steps

- Suppose that $\underline{u}_T, \underline{u}_B, \underline{u}_I$ are the vectors of top, bottom, and internal nodal DoFs.
- Then the governing equations look like:

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ \underline{\underline{M}}_{BT} & \underline{\underline{M}}_{BB} & \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \ddot{\underline{u}}_T \\ \ddot{\underline{u}}_B \\ \ddot{\underline{u}}_I \end{bmatrix} + \begin{bmatrix} \underline{\underline{K}}_{TT} & \underline{\underline{K}}_{TB} & \underline{\underline{K}}_{TI} \\ \underline{\underline{K}}_{BT} & \underline{\underline{K}}_{BB} & \underline{\underline{K}}_{BI} \\ \text{sym} & & \underline{\underline{K}}_{II} \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(c)} \\ \underline{F}_B^{(c)} \\ \underline{0} \end{bmatrix} = \underline{F}_e(t)$$

- Under the relative coordinate transformation,

Transformed Equations of Motion

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TT} + \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ & \underline{\underline{M}}_{TT} + \underline{\underline{M}}_{BB} + \underline{\underline{M}}_{TB} + \underline{\underline{M}}_{BT} & \underline{\underline{M}}_{TI} + \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \ddot{\underline{u}}_T \\ \ddot{\underline{u}}_B \\ \ddot{\underline{u}}_I \end{bmatrix} + \begin{bmatrix} & & \\ & \overline{\underline{\underline{K}}} & \\ & & \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(c)} \\ \underline{0} \\ \underline{0} \end{bmatrix} = \underline{F}_e(t).$$

2.1. Relative Coordinates Pipeline

Outline of the Steps

- Suppose that $\underline{u}_T, \underline{u}_B, \underline{u}_I$ are the vectors of top, bottom, and internal nodal DoFs.
- Then the governing equations look like:

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ \underline{\underline{M}}_{TB}^T & \underline{\underline{M}}_{BB} & \underline{\underline{M}}_{BI} \\ \text{sym} & \underline{\underline{M}}_{BI}^T & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \underline{\ddot{u}}_T \\ \underline{\ddot{u}}_B \\ \underline{\ddot{u}}_I \end{bmatrix} + \begin{bmatrix} \underline{\underline{K}}_{TT} & \underline{\underline{K}}_{TB} & \underline{\underline{K}}_{TI} \\ \underline{\underline{K}}_{TB}^T & \underline{\underline{K}}_{BB} & \underline{\underline{K}}_{BI} \\ \text{sym} & \underline{\underline{K}}_{BI}^T & \underline{\underline{K}}_{II} \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(c)} \\ \underline{F}_B^{(c)} \\ \underline{F}_I^{(c)} \end{bmatrix} = \underline{F}_e(t)$$

Thankfully, this is possible fully in Abaqus itself!

- Under the relative coordinate transformation,

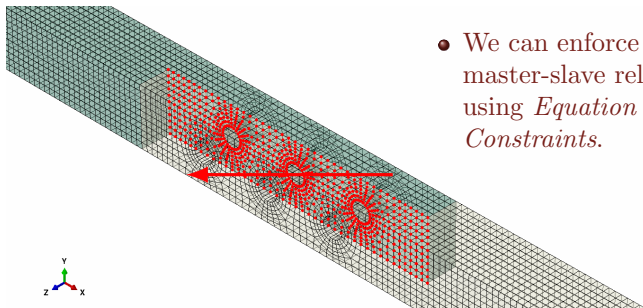
Transformed Equations of Motion

$$\begin{bmatrix} \underline{\underline{M}}_{TT} & \underline{\underline{M}}_{TT} + \underline{\underline{M}}_{TB} & \underline{\underline{M}}_{TI} \\ & \underline{\underline{M}}_{TT} + \underline{\underline{M}}_{BB} + \underline{\underline{M}}_{TB} + \underline{\underline{M}}_{TB}^T & \underline{\underline{M}}_{TI} + \underline{\underline{M}}_{BI} \\ \text{sym} & & \underline{\underline{M}}_{II} \end{bmatrix} \begin{bmatrix} \underline{\ddot{u}}_T \\ \underline{\ddot{u}}_B \\ \underline{\ddot{u}}_I \end{bmatrix} + \begin{bmatrix} \underline{\underline{K}}_{TT} & \underline{\underline{K}}_{TB} & \underline{\underline{K}}_{TI} \\ & \underline{\underline{K}}_{TT} + \underline{\underline{K}}_{BB} + \underline{\underline{K}}_{TB} + \underline{\underline{K}}_{TB}^T & \underline{\underline{K}}_{TI} + \underline{\underline{K}}_{BI} \\ \text{sym} & & \underline{\underline{K}}_{II} \end{bmatrix} \begin{bmatrix} \underline{u}_T \\ \underline{u}_B \\ \underline{u}_I \end{bmatrix} + \begin{bmatrix} \underline{F}_T^{(c)} \\ \underline{F}_B^{(c)} \\ \underline{F}_I^{(c)} \end{bmatrix} = \underline{F}_e(t).$$

2.1. Relative Coordinates Pipeline

Outline of the Steps

- By adding Reference Points to the model at the location of the interfacial nodes, we can create **slave nodes whos DoF will be the relative DoF of the corresponding nodes on the interface.**
- This can be done quite easily with Python scripting (the website has a script for the BRB which can be adapted to arbitrary contexts).



- We can enforce the master-slave relationship using *Equation Constraints*.

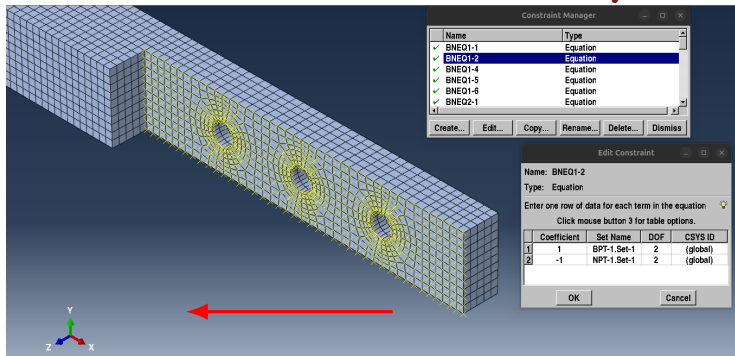
2.1. Relative Coordinates Pipeline

Outline of the Steps

- By adding Reference Points to the model at the location of the interfacial nodes

...And this is what it looks like on ABAQUS.

- The
- SC



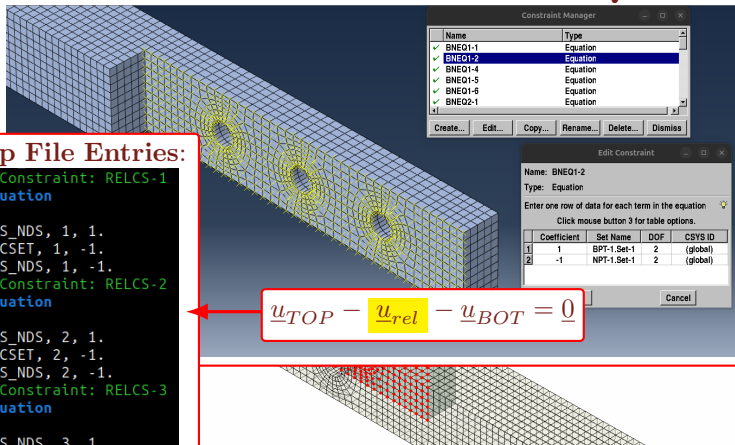
2.1. Relative Coordinates Pipeline

Outline of the Steps

- By adding Reference Points to the model at the location of the interfacial nodes

...And this is what it looks like on ABAQUS.

- Then



.inp File Entries:

```

** Constraint: RELCS-1
*Equation
3
TOPS_NDS, 1, 1.
RELCSSET, 1, -1.
BOTS_NDS, 1, -1.
** Constraint: RELCS-2
*Equation
3
TOPS_NDS, 2, 1.
RELCSSET, 2, -1.
BOTS_NDS, 2, -1.
** Constraint: RELCS-3
*Equation
3
TOPS_NDS, 3, 1.
RELCSSET, 3, -1.
BOTS_NDS, 3, -1.

```

$$\underline{u}_{TOP} - \underline{u}_{rel} - \underline{u}_{BOT} = 0$$

Nonlinear Analysis in Under 100 Lines of MATLAB Code!

3. Nonlinear Analysis in MATLAB

4. Outro and Recommendations

- When we're interested in working on friction modeling at the interfacial level, it makes sense to off-load everything else to a software.
- Try to think of ways to utilize the ABAQUS solvers as much as possible (for CMS modal analysis, etc.).

Some Drawbacks of this Pipeline

- Applicability is **strictly restricted to small displacement contact**.
- Geometrical Nonlinearities can't be present. If so, try an ICE fitting coupled with the relative coordinates approach.

Thank You!



All the detailed instructions are hosted through Github in a repository named [Nidish96/Abaqus4Joints](#)

Comments, suggestions and contributions welcome!
Please email me at nidish@iitm.ac.in.

References I