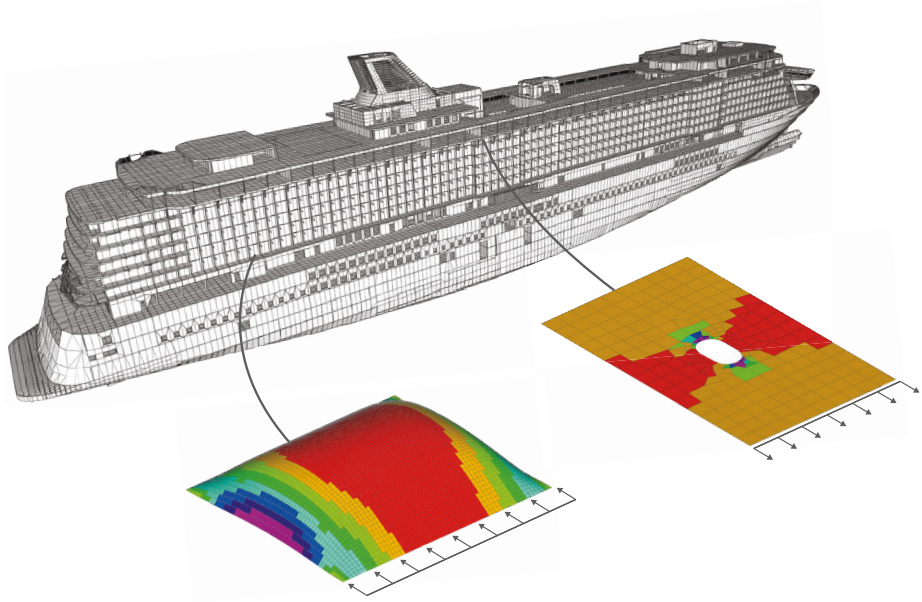
Department of Mechanical Engineering

Abaqus UGENS subroutine for nonlinear analysis of periodic panels

Bruno Reinaldo Goncalves, Jasmin Jelovica,
Jani Romanoff



Aalto University

SCIENCE + TECHNOLOGY

RESEARCH REPORT

Aalto University publication series
SCIENCE + **TECHNOLOGY** 9/2016

Abaqus UGENS subroutine for nonlinear analysis of periodic panels

Bruno Reinaldo Goncalves, Jasmin Jelovica, Jani Romanoff

Aalto University
School of Engineering
Department of Mechanical Engineering
Marine Technology

Aalto University publication series
SCIENCE + **TECHNOLOGY** 9/2016

ISBN 978-952-60-6905-0 (pdf)
ISSN-L 1799-4896
ISSN 1799-4896 (printed)
ISSN 1799-490X (pdf)
http://urn.fi/URN:ISBN:978-952-60-6905-0

Unigrafia Oy
Helsinki 2016

Finland

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Bruno Reinaldo Goncalves, Jasmin Jelovica, Jani Romanoff

Name of the publication
Abaqus UGENS subroutine for nonlinear analysis of periodic panels

Publisher School of Engineering

Unit Department of Mechanical Engineering

Series Aalto University publication series SCIENCE + TECHNOLOGY 9/2016

Field of research Marine Technology

Abstract

This report describes an Abaqus UGENS subroutine for geometric and material nonlinear analysis of periodic panels using the first-order shear deformation theory. The structure is modelled with shell elements, as one layer of equivalent mechanical properties. The subroutine modifies the stiffness matrix of each shell element of the mesh separately based on its strain state. It relies on pre-computed stiffness curves that define the ABCD stiffness matrix of a unit cell. By looking at combinations of force and strain, the code interpolates the stiffness curves to calculate equivalent nonlinear stiffness. Complex stress states with different types of nonlinearity occurring simultaneously in the structure can be described. The examples show that the subroutine can deal with nonlinearities such as global buckling, local buckling and post-yield response with good accuracy and low computational cost compared to conventional FEM. The report includes the necessary information to set up the subroutine, including selection and compatibility of software and packages and input file preparation. Web-core sandwich panels are used as example throughout the report; the same principles are valid for any periodic structure. The full implementation is given in Appendix.

Keywords UGENS subroutine, Homogenization, Sandwich structure, Periodic structure, Multiscale modelling, Nonlinear analysis, Ultimate strength

ISBN (printed)	**ISBN (pdf)** 978-952-60-6905-0	
ISSN-L 1799-4896	**ISSN (printed)** 1799-4896	**ISSN (pdf)** 1799-490X
Location of publisher Helsinki	**Location of printing** College Station TX, USA	**Year** 2016
Pages	**urn** http://urn.fi/URN:ISBN:978-952-60-6905-0	

Acknowledgements

The authors would like to gratefully acknowledge the support of Finland Distinguished Professor (FiDiPro) project "Non-linear Response of Large, Complex Thin-Walled Structures", funded by the Finnish Funding Agency for Innovation (Tekes), Deltamarin, Napa Ltd, Koneteknolo-giakeskus Turku, Ruukki and Meyer Turku Oy. We would also like to thank *CSC – IT Centre for Science Ltd.* for providing Abaqus™ package license.

Contents

1. Introduction

Structural core sandwich panels are a current alternative for lightweight structures, offering efficiency in terms of stiffness and integration of non-structural functions in tight spaces. Due to the different scales involved, modelling sandwich panels with conventional finite element models is very time- and resource-consuming. Dozens of such models might be needed to understand the failure path or optimize a large structure, making the process inefficient from the industry's perspective. Therefore, simplifications are needed to improve efficiency and safety, even if they only give an approximate description of the deformation physics; see Reinaldo Goncalves et al. (2016) for a short summary on modelling alternatives.

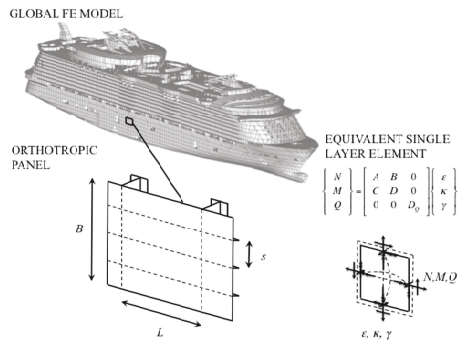


Figure 1. Equivalent single layer theory for analysis of large structures. Reproduced from Romanoff et al. (2016).

Using the equivalent single layer (ESL) theory with first-order kinematics (fig.1) is one of the most convenient approaches in this sense; see the review article by Romanoff et al. (2016). It is relatively simple, very resource-efficient and can be implemented in commercial finite element platforms. Jelovica and Romanoff (2013) showed that the ESL theory can be used to accurately describe global buckling of web-core sandwich panels. Reinaldo Goncalves et al. (2016) extended it to capture geometric nonlinear effects at the unit cell level, such as local buckling.

The objective of this work is to present an Abaqus user subroutine UGENS for geometric and material nonlinear analysis of periodic structures. It allows for different shell behaviour in tension and compression. The stiffness matrix

of each shell element in the panel domain is based on its strain state, according to RVE analysis. Therefore the method can deal with localized nonlinearity (for example local yielding or buckling near panel boundaries). Another implication is that different nonlinear effects can occur simultaneously in the panel domain. The subroutine is presented in this report for web-core sandwich panels, yet it can be extended to any periodic structure by modifying certain input arrays. This might include different types of composites or metal stiffened panels. See Reinaldo Goncalves et al. (2016) for extension to corrugated-type panels.

The subroutine takes stress resultant vs. strain curves derived from RVE analyses (*stiffness curves*) and modifies the stiffness of each shell element based on its strain state:

$$(N_i, M_i) = \{[\mathrm{A}], [\mathrm{B}], [\mathrm{C}], [\mathrm{D}]\}_{ij,RVE}(\varepsilon_j, \lambda_j), \qquad i,j = 1..6 \tag{1}$$

For every FE iteration, the shell element's strain components are located in the stiffness curves to define the equivalent stiffness matrix, update forces and moments. In the subroutine here presented, the stiffness curves are defined according to Reinaldo Goncalves et al. (2016) in compression and Korgesaar et al. (2016) in tension. In compression, no material nonlinearity is considered at this point. The method shown in Reinaldo Goncalves et al. (2016) is also suitable to estimate tensile behaviour, yet the analytical solution of Korgesaar et al. (2016) is used for simplicity.

In the following sections, the code and input file preparation are explained step-by-step.

2. Methods

2.1 Abaqus UGENS subroutine overview

The Abaqus UGENS subroutine is used to modify the mechanical behaviour of shell elements in terms of generalized quantities (Simulia, 2013). In essence, it makes possible to define the relations between strain and generalized forces for each load increment as in eq. (1). Convergence parameters such as the Jacobian and other state variables can also be updated.

2.2 Definition of stiffness curves for web-core panel

A RVE that includes local panel imperfections is defined according to Reinaldo Goncalves et al. (2016) to describe the geometric nonlinear behaviour of the unit cell in compression. Enforced displacement boundary conditions are applied following the strain field of the first-order shear deformation theory. That is, uniform displacements and curvatures in X- and Y- directions, shear and torsion, see fig.2. Periodicity is maintained in the RVE edges perpendicular to loading, while the RVE is free-free in the thickness direction.

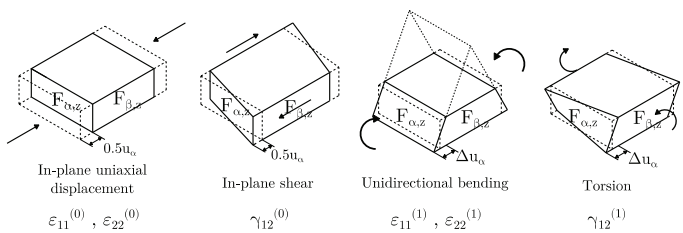


Figure 2. Boundary conditions used in the RVE models. Reproduced from Reinaldo Goncalves et al. (2016).

Six RVE models are necessary, one for each strain component of fig.2. Geometric nonlinear analysis of the RVE is then performed. Based on the RVE edge external forces, equivalent generalized shell resultants are estimated; see Reinaldo Goncalves et al. (2016). The stress resultants obtained for web-core RVE with imperfections are reproduced in fig.3. Material nonlinearity in compression is not considered in this case.

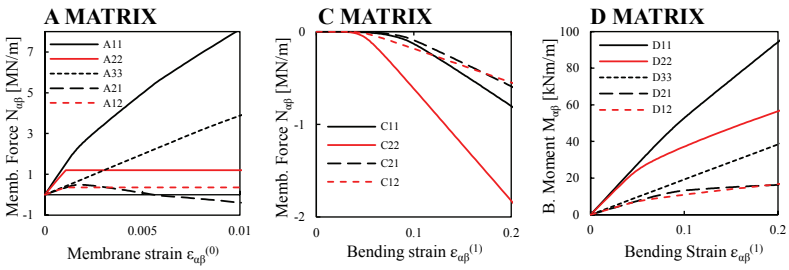


Figure 3. Stiffness curves (stress resultant vs. strain relations) of a web-core sandwich panel in compression. Reproduced from Reinaldo Goncalves et al. (2016).

In tension, the stiffness curves for material nonlinear behaviour are determined based on analytical expressions. A_{11} and A_{22} are defined assuming plane strain tension state according to Korgesaar et al. (2016). If the shell element is yielding, it is assumed that the whole stiffness matrix, except shear and torsional terms, is reduced proportionally to A_{11} or A_{22}, depending on the critical direction, that is

$$\begin{aligned} ([A],[B],[C],[D])_p &= ([A],[B],[C],[D])_e \cdot \frac{A_{11,p}}{A_{11,e}}, \quad e_1 > e_{1,p} \\ ([A],[B],[C],[D])_p &= ([A],[B],[C],[D])_e \cdot \frac{A_{22,p}}{A_{22,e}}, \quad e_2 > e_{2,p} \end{aligned} \tag{2}$$

where the subscripts $_e$ and $_p$ correspond to elastic and plastic values respectively. The same assumption is made in the application examples of Korgesaar et al. (2016). Transverse shear stiffness is assumed to be constant with its linear elastic analytical magnitude in both, tension and compression. The subroutine UGENS does not allow it to be updated during the analysis.

2.3 Stiffness curves input in the user subroutine

The *stiffness curves* (stress resultant vs. strain) are interpolated into predefined strain points for code input. The pre-defined strains are shown from lines 29-58 of the code (fig.4)

```
C STRESS RESULTANT vs. STRAIN RELATIONS / RVE ANALYSIS ----------------------------------------

C 1.PRE-DEFINED STRAIN INTERVALS

      eA11=(/-2.000e+00,-1.000e+00,-5.000e-01,-2.950e-01,-2.850e-01,
     + -2.750e-01,-2.650e-01,-2.550e-01,-2.450e-01,-2.350e-01,
     + -2.250e-01,-2.150e-01,-2.050e-01,-1.950e-01,-1.850e-01,
     + -1.750e-01,-1.650e-01,-1.550e-01,-1.450e-01,-1.350e-01,
     + -1.250e-01,-1.150e-01,-1.050e-01,-9.500e-02,-8.500e-02,
     + -7.500e-02,-6.500e-02,-5.500e-02,-4.500e-02,-3.500e-02,
     + -2.500e-02,-1.500e-02,-1.400e-02,-1.300e-02,-1.200e-02,
     + -1.100e-02,-1.000e-02,-9.000e-03,-8.000e-03,-7.000e-03,
     + -6.000e-03,-5.500e-03,-5.000e-03,-4.500e-03,-4.000e-03,
     + -3.500e-03,-3.000e-03,-2.500e-03,-2.000e-03,-1.500e-03,
     + -1.000e-03,-5.000e-04,0.000e+00,5.000e-04,1.000e-03,
     + 1.400e-03,1.450e-03,1.475e-03,1.500e-03,1.525e-03,
     + 1.550e-03,1.600e-03,2.000e-03,2.500e-03,3.000e-03,
     + 3.500e-03,4.000e-03,4.500e-03,5.000e-03,5.500e-03,
     + 6.000e-03,7.000e-03,8.000e-03,9.000e-03,1.000e-02,
```

```
+ 1.100e-02,1.200e-02,1.300e-02,1.400e-02,1.500e-02,
+ 2.500e-02,3.500e-02,4.500e-02,5.500e-02,6.500e-02,
+ 7.500e-02,8.500e-02,9.500e-02,1.050e-01,1.150e-01,
+ 1.250e-01,1.350e-01,1.450e-01,1.550e-01,1.650e-01,
+ 1.750e-01,1.850e-01,1.950e-01,2.050e-01,2.150e-01,
+ 2.250e-01,2.350e-01,2.450e-01,2.550e-01,2.650e-01,
+ 2.750e-01,2.850e-01,2.950e-01,5.000e-01,1.000e+00,
+ 2.000e+00/)

eA12=eA11; eA21=eA11; eA22=eA11; eA33=eA11; eD11=eA11; eD12=eA11;
eD21=eA11; eD22=eA11; eD33=eA11; eC11=eA11; eC12=eA11; eC21=eA11;
eC22=eA11; eC33=eA11
```

Figure 4. Definition of pre-defined *strain points*

The points are taken arbitrarily and are the same for all stiffnesses in the sample code; the number of points must be sufficient to describe the original stiffness curves. The interpolated forces and moments corresponding to each strain point are shown in lines 59-398. For example, the force N_1 taken from the model for displacement u_1 in fig.2 is defined as FA11 (defines A_{11} stiffness in conjunction with the corresponding strain) as shown in fig.5.

```
C 2. CORRESPONDING FORCES

 FA11=(/-1.150e+09,-5.762e+08,-2.893e+08,-1.716e+08,-1.659e+08,
+ -1.602e+08,-1.544e+08,-1.487e+08,-1.429e+08,-1.372e+08,
+ -1.315e+08,-1.257e+08,-1.200e+08,-1.143e+08,-1.085e+08,
+ -1.028e+08,-9.704e+07,-9.131e+07,-8.557e+07,-7.983e+07,
+ -7.409e+07,-6.835e+07,-6.262e+07,-5.688e+07,-5.114e+07,
+ -4.540e+07,-3.966e+07,-3.392e+07,-2.819e+07,-2.245e+07,
+ -1.671e+07,-1.097e+07,-1.040e+07,-9.824e+06,-9.250e+06,
+ -8.676e+06,-8.102e+06,-7.522e+06,-6.927e+06,-6.318e+06,
+ -5.691e+06,-5.346e+06,-4.968e+06,-4.589e+06,-4.205e+06,
+ -3.817e+06,-3.423e+06,-3.019e+06,-2.591e+06,-2.077e+06,
+ -1.419e+06,-7.107e+05,0.0000e+00,7.0327e+05,1.4065e+06,
+ 1.9691e+06,2.0395e+06,
+ 2.0641e+06,2.0764e+06,2.0789e+06,2.0798e+06,2.0814e+06,
+ 2.0911e+06,2.1027e+06,2.1142e+06,2.1253e+06,2.1363e+06,
+ 2.1470e+06,2.1575e+06,2.1678e+06,2.1779e+06,2.1975e+06,
+ 2.2164e+06,2.2346e+06,2.2522e+06,2.2692e+06,2.2856e+06,
+ 2.3015e+06,2.3170e+06,2.3319e+06,2.4605e+06,2.5610e+06,
+ 2.6421e+06,2.7090e+06,2.7651e+06,2.8126e+06,2.8530e+06,
+ 2.8877e+06,2.9174e+06,2.9430e+06,2.9650e+06,2.9839e+06,
+ 3.0001e+06,3.0138e+06,3.0254e+06,3.0351e+06,3.0431e+06,
+ 3.0496e+06,3.0546e+06,3.0585e+06,3.0612e+06,3.0629e+06,
+ 3.0636e+06,3.0636e+06,3.0627e+06,3.0611e+06,3.0589e+06,
+ 3.0560e+06,2.9225e+06,2.4819e+06,1.8434e+06/)
```

Figure 5. Interpolated forces from the *stiffness curves* (FA11)

The code then creates stiffness and strain three-dimensional arrays (lines 451-505). In both arrays the first index is the ID: we define ID as a counter running from one to the number of strain points. The second and third indices define the components in the 6x6 stiffness matrix. In short:

STIF(ID,i,j) = stiffness K(i,j) at a given ID

e(ID,i,j) = strain ε(i,j) at a given ID

STIF(*ID,I,j*) and **e**(ID,i,j) have one to one pairing; each stiffness component at a given ID has a corresponding strain. Figure 6 shows the three-dimensional arrays assembling process.

```
C ASSEMBLE STIFFNESS 3D ARRAY

do m=2,111
  STIF(m-1,1,1) = (FA11(m)-FA11(m-1))/(eA11(m)-eA11(m-1))
  STIF(m-1,1,2) = (FA12(m)-FA12(m-1))/(eA12(m)-eA12(m-1))
  STIF(m-1,1,4) = (FC11(m)-FC11(m-1))/(eC11(m)-eC11(m-1))
  STIF(m-1,1,5) = (FC12(m)-FC12(m-1))/(eC12(m)-eC12(m-1))
  STIF(m-1,2,1) = (FA21(m)-FA21(m-1))/(eA21(m)-eA21(m-1))
  STIF(m-1,2,2) = (FA22(m)-FA22(m-1))/(eA22(m)-eA22(m-1))
  STIF(m-1,2,4) = (FC21(m)-FC21(m-1))/(eC21(m)-eC21(m-1))
  STIF(m-1,2,5) = (FC22(m)-FC22(m-1))/(eC22(m)-eC22(m-1))
  STIF(m-1,3,3) = (FA33(m)-FA33(m-1))/(eA33(m)-eA33(m-1))
  STIF(m-1,4,4) = (MD11(m)-MD11(m-1))/(eD11(m)-eD11(m-1))
  STIF(m-1,4,5) = (MD12(m)-MD12(m-1))/(eD12(m)-eD12(m-1))
  STIF(m-1,5,4) = (MD21(m)-MD21(m-1))/(eD21(m)-eD21(m-1))
  STIF(m-1,5,5) = (MD22(m)-MD22(m-1))/(eD22(m)-eD22(m-1))
  STIF(m-1,6,6) = (MD33(m)-MD33(m-1))/(eD33(m)-eD33(m-1))
end do
```

(a)

```
C ASSEMBLE STRAIN 3D ARRAY

do m=2,111
  e(m-1,1,1) = eA11(m)
  e(m-1,2,1) = eA21(m)
  e(m-1,4,1) = eB11(m)
  e(m-1,5,1) = eB21(m)
  e(m-1,1,2) = eA12(m)
  e(m-1,2,2) = eA22(m)
  e(m-1,4,2) = eB12(m)
  e(m-1,5,2) = eB22(m)
  e(m-1,3,3) = eA33(m)
  e(m-1,6,3) = eB33(m)
  e(m-1,1,4) = eC11(m)
  e(m-1,2,4) = eC21(m)
  e(m-1,4,4) = eD11(m)
  e(m-1,5,4) = eD21(m)
  e(m-1,1,5) = eC12(m)
  e(m-1,2,5) = eC22(m)
  e(m-1,4,5) = eD12(m)
  e(m-1,5,5) = eD22(m)
  e(m-1,3,6) = eC33(m)
  e(m-1,6,6) = eD33(m)
end do
```

(b)

Figure 6. (a) Stiffness and (b) strain three-dimensional arrays according to the *stiffness curves*

Stiffness is calculated between the strain points < *strain-1* > and < *strain* >. Therefore, both matrices start from the second pre-defined strain point. A given strain ID points to the stiffness at its left hand side: fig.7 shows an example of stiffness definition based on strain points and corresponding forces.

2.4 Update of section forces

In this section, the shell element stiffness matrix update algorithm is explained (lines 581-729). For every strain component in every element, the Abaqus-determined current strain ε_j and next strain $\varepsilon_j+d\varepsilon_j$ are located in the corresponding **e**(ID,i,j) strain array; ID(ε_j) and ID($\varepsilon_j+d\varepsilon_j$) are obtained . Based on the IDs, the stiffness matrix can be determined from **STIF**(*ID,I,j*). Interpolation is needed if the IDs change with the strain increment. A closer look on the interpolation process follows.

For every finite element, every strain increment, every iteration, a loop runs from (*i = 1 to 6*) and (*j = 1 to 6*). The index *i* loops among the force components, while *j* loops among strain components. That is,

$$(N_i) = \{STIF_{ID,i,j}\}(\varepsilon_j) \tag{3}$$

The next step is to determine, for every combination (*i,j*), where the current strain and increment are in **STIF**(*ID,I,j*). In the sample code for web-core sandwich panels, 110 IDs define strain and stiffness three-dimensional arrays. The start ID (in the example provided) is either 52 or 51 for positive or negative strain component respectively, see fig.7.

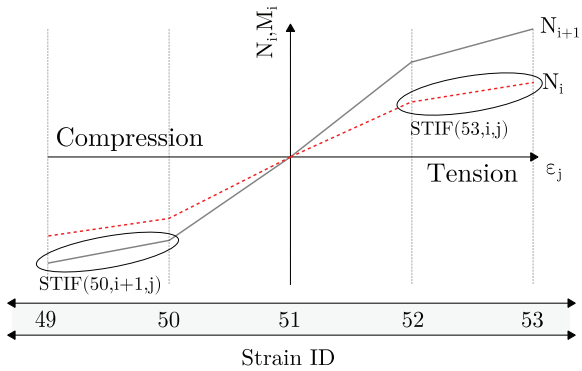


Figure 7. Determination of stiffness based on *stiffness curves* N_i vs ε_j and N_{i+1} vs ε_j

In lines 525-585, the strain IDs based on current strain and increment are calculated (fig.8)

```
C Check current strain ID
C Start ID = 53 if tension / 52 if compression

      curr_ID=53
      ii=53

11     if (curr_ID.EQ.53) then
         stranCurrent=stran(j)
      if (stranCurrent .EQ. 0.0) then
         if (dstran(j) .LT. 0.0) then
           curr_ID=52
         end if
      else if (stranCurrent .GT. 0.0) then
        if (e(ii,i,j).GT.stranCurrent) then
          curr_ID=ii
        else
          ii=ii+1
          goto 11
        end if
      else if (stranCurrent .LT. 0.0) then
         if (e((ii-1),i,j).LT.stranCurrent) then
           curr_ID=ii
         else
           ii=ii-1
           goto 11
         end if
      else
      endif
      endif
```

(a)

```
C Check next strain ID

      next_ID=53
      jj=53

12     if (next_ID.EQ.53) then
         stranNext=stran(j)+dstran(j)
      if (stranNext .GT. 0.0) then
         if (e(jj,i,j).GT.stranNext) then
           next_ID=jj
         else
           jj=jj+1
           goto 12
         end if
      else if (stranNext .LT. 0.0) then
         if (e((jj-1),i,j).LT.stranNext) then
           next_ID=jj
         else
           jj=jj-1
           goto 12
         end if
      endif
      endif
```

(b)

Figure 8. Determination of strain ID according to strain, increment and strain array e(ID,i,j)

The help variable n checks in how many strain ID intervals the increment moves in relation to the current strain

$$n = next\ ID - current\ ID = ID(\varepsilon_j + d\varepsilon_j) - ID(\varepsilon_j) \tag{4}$$

First, we check if strain and increment have same sign. That is, if the strain is increasing or decreasing in magnitude. Let the strain increase monotonically

for the cases discussed below. In the code, both cases are treated following the same principles.

In the simplest case, strain and increment are in the same interval (*n=0*), as in fig.9. The force update is given by

$$N_i = N_i + STIF(ID, i, j)d\varepsilon_j \tag{5}$$

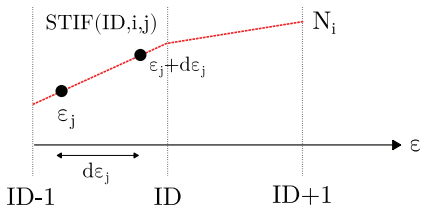


Figure 9. Stiffness update, *n=0*

If ε_j and $\varepsilon_j+d\varepsilon_j$ are not in the same ID interval, $n \neq 0$ and interpolation is needed to determine the section force change. In the cases when $n=\pm 1$, that is, when strain and increment are one step apart, the force update becomes (fig.10):

$$n = 1 \;\rightarrow N_i = N_i + STIF(ID, i, j)(\varepsilon_{ID} - \varepsilon_j) + STIF\big((ID+1), i, j\big)(\varepsilon_j + d\varepsilon_j - \varepsilon_{ID}) \tag{6}$$

$$n = -1 \;\rightarrow N_i = N_i + STIF(ID, i, j)(\varepsilon_j - \varepsilon_{ID-1}) + STIF\big((ID-1), i, j\big)(\varepsilon_j + d\varepsilon_j - \varepsilon_{ID-1}) \tag{7}$$

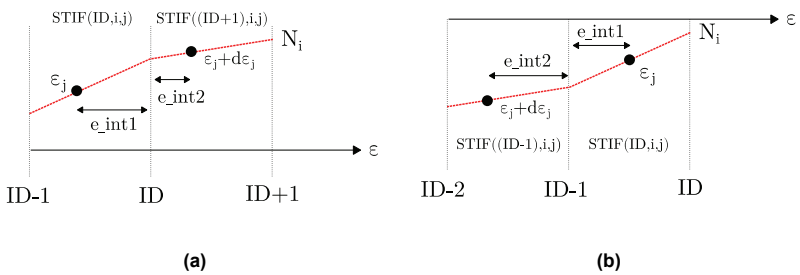


Figure 10. Stiffness update, (a) *n=1 (b) n=-1*

Now, let $n > 1$, that is, the strain increment increases monotonically by more than one ID. We follow the previous case, adding the force that corresponds to the intermediate interval.

$$n > 1 \;\rightarrow N_i = N_i + STIF(ID, i, j)(\varepsilon_{ID} - \varepsilon_j) + STIF\big((ID+1), i, j\big)(\varepsilon_{ID+1} - \varepsilon_{ID}) + STIF\big((ID+2), i, j\big)(\varepsilon_j + d\varepsilon_j - \varepsilon_{ID+1}) \tag{8}$$

Figure 11 illustrates a sample case, $n = 2$.

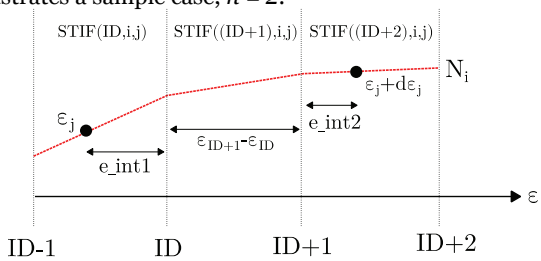


Figure 11. Stiffness update, *n=2*

A specific procedure is used for tensile behaviour in the current code (fig.12) from lines 596-650. The stiffnesses A_{11} and A_{22} are calculated analytically as in Korgesaar et al. (2016) assuming plane strain tension state, and used to update the whole stiffness matrix approximately. It is assumed that extensional, bending and extensional-bending stiffnesses in the directions *1* and *2* are reduced after yielding in the same proportion as A_{11} or A_{22}, according to the most stressed direction. The reduction is made in terms of the theoretical stiffness values given in the input file (PROPS) and a factor *k_nl* (fig.12), which relates nonlinear and linear extensional stiffnesses A_{11} and A_{22}.

```
C Reduce STIF(ID,i,j) if shell yielding in 11 or 22 directions
    ij=(i-1)*6+j

    if (stran(1).GE.1.478e-3) then
     if (stran(1).GT.stran(2)) then
      if (ij.EQ.1) then
           A11nonl=STIF(curr_ID,i,j)
           k_nl=A11nonl/PROPS(1)
      else
      select case (ij)
       case (2);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
       case (4);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(7)*k_nl
       case (5);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(11)*k_nl
       case (7);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
       case (8);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(3)*k_nl
       case (10);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(8)*k_nl
       case (11);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(12)*k_nl
       case (15);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(6)
       case (22);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(10)*k_nl
       case (23);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
       case (28);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
       case (29);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(15)*k_nl
       case (36);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(21)
       case default; curr_ID=1;n=0;STIF(curr_ID,i,j)=0.0
      end select
     end if
    end if
   end if
```

(a)

```
if (stran(2).GE.1.458e-3) then
 if (stran(2).GT.stran(1)) then
  if (ij.EQ.8) then
       A22nonl=STIF(curr_ID,i,j)
       k_nl=A22nonl/PROPS(3)
  else
   select case (ij)
    case (1);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(1)*k_nl
    case (2);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
    case (4);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(7)*k_nl
    case (5);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(11)*k_nl
    case (7);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
    case (10);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(8)*k_nl
    case (11);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(12)*k_nl
    case (15);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(6)
    case (22);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(10)*k_nl
    case (23);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
    case (28);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
    case (29);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(15)*k_nl
    case (36);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(21)
    case default; curr_ID=1;n=0;STIF(curr_ID,i,j)=0.0
   end select
  end if
 end if
end if
```

(b)

Figure 12. Stiffness matrix update for yielding shell element (a) main direction ε_{11} (b) main direction ε_{22}

In shear and torsion (A_{33} and D_{33}), the linear elastic stiffness value is used in tension overall. Future work is needed to accurately determine the unit cell behaviour and equivalent resultants under bending and shear after the yield point.

2.5 State variable update

In the code proposed, no state variable except of FORCE(i) is updated. The Jacobian is defined based on the analytical linear elastic stiffness matrix. The Jacobian affects only the convergence rate and such approximation was observed to be adequate.

3. Abaqus preparation

3.1 Input file

The input file must be specifically prepared for using the UGENS subroutine. The shell elements that follow the subroutine must be defined using the shell general section command:

> *SHELL GENERAL SECTION, ELSET=name_of_element_set, USER, VARIABLES=number_state_var, PROPERTIES=21 (or 36 if the stiffness matrix is unsymmetrical).

After that, the shell thickness (taken 0.045m for the web-core sandwich panel studied as the distance between outer surfaces of the face plates) and initial stiffness matrix (e.g. in the elastic regime) must be given. If 21 properties are chosen, only the symmetric part is given following the order shown in fig.13.

$$\begin{bmatrix} A_{11} & A_{12} & 0 & C_{11} & C_{12} & 0 \\ A_{21} & A_{22} & 0 & C_{21} & C_{22} & 0 \\ 0 & 0 & A_{33} & 0 & 0 & C_{33} \\ B_{11} & B_{12} & 0 & D_{11} & D_{12} & 0 \\ B_{21} & B_{22} & 0 & D_{21} & D_{22} & 0 \\ 0 & 0 & B_{33} & 0 & 0 & D_{33} \end{bmatrix} \begin{matrix} 1 & 2 & 4 & 7 & 11 & 16 \\ & 3 & 5 & 8 & 12 & 17 \\ & & 6 & 9 & 13 & 18 \\ & & & 10 & 14 & 19 \\ & & & & 15 & 20 \\ & & & & & 21 \end{matrix}$$

Figure 13. Stiffness property order for input file (symmetric part of the stiffness matrix)

The keyword *TRANSVERSE SHEAR STIFFNESS must be used, followed by the three linear elastic transverse shear stiffness values. One example of the main commands used with the UGENS subroutine is shown in fig.14.

```
*SHELL GENERAL SECTION, ELSET=P1, USER,VARIABLES=60,PROPERTIES=21
    0.045,
.140653E+10,  0.339560E+09,  0.113187E+10,  0.000000E+00,  0.000000E+00,  0.396154E+09,  0.186265E-08,  0.465661E-09
.000000E+00,  0.547731E+06,  0.465661E-09,  0.186265E-08,  0.000000E+00,  0.153333E+06,  0.511109E+06,  0.000000E+00
.000000E+00,  0.000000E+00,  0.000000E+00,  0.000000E+00,  0.178888E+06
*TRANSVERSE SHEAR STIFFNESS
.679631E+08,  0.418832E+06,  0.000000E+00
*IMPERFECTION, FILE=gnPlate3mB, STEP=1
  1,  0.003
** Load Step 2 ----------------------------------------------------------
*STEP, NLGEOM, INC=130
Incremental
*STATIC, RIKS
      .001, 1., , 0.001, 0.018, , ,
```

Figure 14. Example of input file for UGENS subroutine

The input file should be otherwise prepared as for standard geometric nonlinear analysis.

3.2 Subroutine call

The subroutine is run using the Abaqus batch call, for example:

```
abq6133 job=file_name user=subroutine_name
```

where *abq6133* corresponds to the Abaqus version 6.13-3. The subroutine file must be in the Abaqus scratch folder (typically C:\TEMP).

3.3 Obtaining a converged solution

As the method here described is aimed to structures that exhibit considerable nonlinear effects such as buckling, it is often appropriate to use the Riks method to trace the nonlinear path. The Modified Riks Method can deal with unstable response, such as snap-through and snap-back, passing limit points by modifying the Newton's method with an additional load proportionality factor. Convergence difficulties might be observed if dramatic changes of mechanical properties are present between steps or among elements. Some suggestions to deal with possible convergence problems are listed below:

- To decrease maximum load step size, inducing the FE package not to overshoot the next stiffness estimate
- To smooth out sudden stiffness changes
- To supplement the input file with user-defined Abaqus control parameters, for instance increasing the allowable number of iterations. For example:

```
*CONTROLS, PARAMETERS=TIME INCREMENTATION
16 , 18 , 20 , 100 , 30 ,  6
```

- Relaxing convergence parameters using the *CONTROL command (care must be taken), for instance:

```
*CONTROLS, PARAMETERS=FIELD, FIELD=DISPLACEMENT
0.02   ,  0.05 ,       ,       , 0.10   ,
```

The meaning of each position under the *CONTROLS keyword is described in Simulia (2013).

3.4 List of software and packages

The following software and packages are used for subroutine development and application:

SIMULIA Abaqus™ 6.13

Microsoft Visual Studio 2008

Intel Fortran Composer XE

See https://software.intel.com/sites/default/files/managed/88/99/Link%20Instructions.pdf for guidance on software compatibility and installation instructions.

4. Examples

In this section, examples that use the code supplemented with web-core sandwich panel stiffnesses are briefly shown. In all analysis enforced displacements are applied at the plate edges. The unit cell dimensions are given in fig.15; the material Young's modulus is 206GPa and Poisson's ratio 0.3.

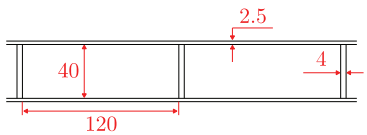


Figure 15. Unit cell geometry in millimeters, taken for all examples. Reproduced from Reinaldo Goncalves et al. (2016)

4.1 Simply supported 3x3m panel in uniaxial compression

The first example refers to the 3x3m simply supported panel shown in Reinaldo Goncalves et al. (2016). The panel is subjected to uniaxial compressive displacement in the web direction and opposite panel edges are allowed to contract in-plane. An initial imperfection based on the plate's first eigenmode is enforced to activate the out-of-plane deformations and promote global buckling. See Reinaldo Goncalves et al. (2016) for more detailed description and analysis of results.

Figure 16 shows very good agreement between the multiscale ESL method and conventional finite element model discretized with shell elements. Buckling is observed first in the panel and later in the unit cell level; local buckling is described thanks to the stiffness curves. Overall, convergence is obtained with a maximum of 3-4 iterations per load increment. The scaled deformed shapes obtained with both methods are shown in fig.17.

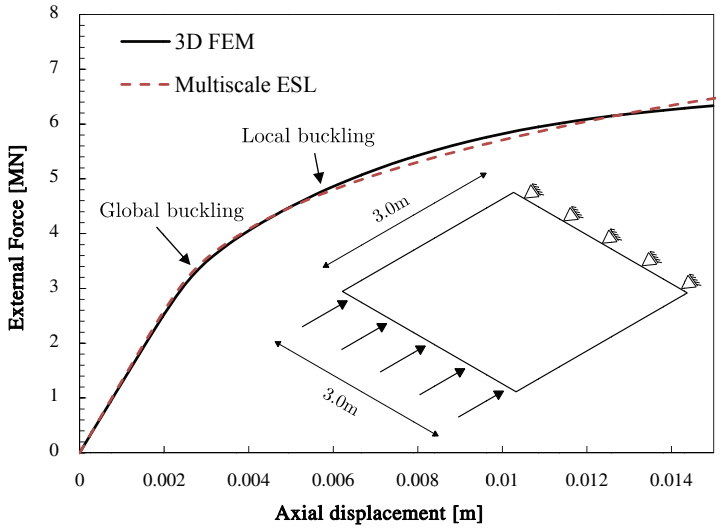


Figure 16. Load vs. displacement curve for 3x3m panel in uniaxial compression and validation against 3D FEM

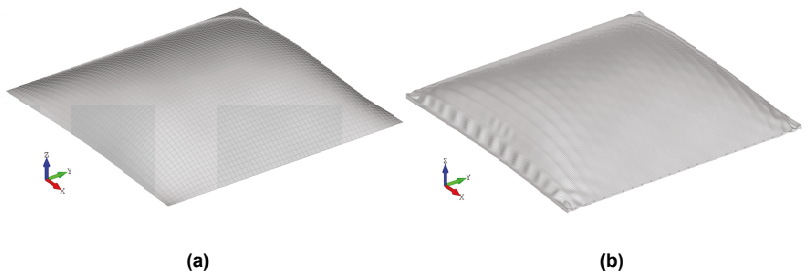


Figure 17. Deformed shapes (5x, axial displacement ≈ 0.13m) for plate in uniaxial tension according to (a) ESL method (b) 3D FEM

4.2 Plane-strain panel with a circular hole in uniaxial tension

In this example, a rectangular plate of dimensions 8.0x4.8m with a circular hole, D=1.0m, at mid-plate is subjected to uniform tensile displacement in its shorter edge. The plate is assumed to be in plane-strain parallel to its longer dimension. The hole introduces stress concentration near its borders, increasing the complexity of the analysis. Convergence is slower than in the compression case of 4.1 due to the dramatic changes in stiffness and highly varying stiffness among elements of the mesh. Figure 18 shows the *load vs. displacement* curve obtained for this example and comparison with an equivalent 3D FEM model.

Very good agreement between 3D and ESL curves is seen in fig.18. The ESL is slightly stiffer after the yield point; the maximum relative difference is approximately 4.5%. The difference is mainly caused by the plane-strain assump-

tion of the ABD matrix analytical derivation, which is violated near the hole as it allows some contraction perpendicular to loading. As this is the most stressed region, some difference in the response is seen. Figure 19 shows the Membrane Force distribution in the web-direction for an axial displacement of 0.3m using both, ESL and 3D FEM. In 3D FEM, the face plate is expected to carry 39% of the membrane force; therefore, the contour is scaled accordingly.

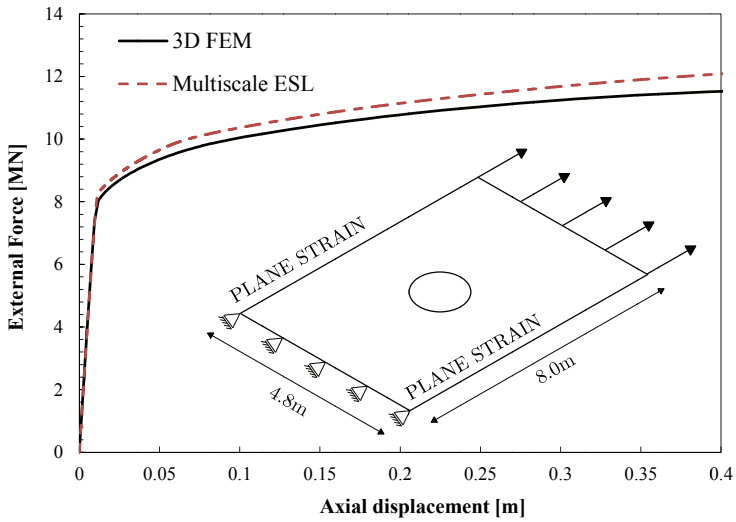


Figure 18. Load vs. displacement curve for 8.0x4.8m panel with circular opening under tensile load and validation against 3D FEM

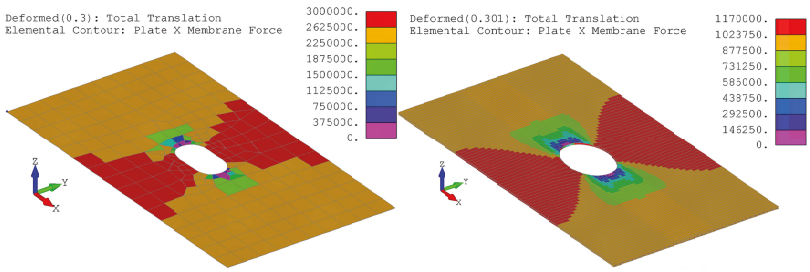


Figure 19. Membrane force distribution in the web direction – axial displacement approximately 0.3m

It can be observed in fig.19 that, despite the ABD estimate assumptions and coarse mesh size, the ESL method is able to describe membrane force distribution in the panel very consistently when compared to the 3D FEM equivalent. The method is therefore seen as suitable for the analysis of large structures, providing a good trade-off between accuracy, simplicity and computational cost.

References

Jelovica, J., Romanoff, J., (2013). *Load-carrying behaviour of web-core sandwich plates in compression*. Thin-Walled Structures, 73, pp.264–272.

Korgesaar, M., Reinaldo Goncalves, B, Romanoff, J., Remes, H. (2016). *Tensile instability of orthotropic web-core sandwich panels.* International Journal of Solids and Structures. Under review.

Reinaldo Goncalves, B., Jelovica, J., Romanoff, J. (2016). *A homogenization method for geometric nonlinear analysis of sandwich structures with initial imperfections.* International Journal of Solids and Structures, 87, pp.194–205.

Romanoff, J., Jelovica, J., Avi, E., Reinaldo Goncalves, B., Korgesaar, M., Raikunen, J., Remes, J., Niemela, A., Reddy, J.N., Varsta, P. (2016). *Use of equivalent single layer plate theory in ship structural design*. In Proceedings of PRADS.

Simulia. (2013) *Abaqus user subroutine reference guide – Abaqus 6.13 version.*

Appendix: UGENS implementation

```
      subroutine UGENS(ddndde,force,statev,sse,spd,pnewdt,stran,dstran,
     1 tss,time,dtime,temp,dtemp,predef,dpred,cename,ndi,nshr,nsecv,
     2 nstatv,props,jprops,nprops,njprop,coords,celent,thick,dfgrd,curv,
     3 basis,noel,npt,kstep,kinc,kit,linper)
c
      include 'aba_param.inc'
c
      character*80 cename
      dimension force(nsecv),statev(nstatv),ddndde(nsecv,nsecv),
     1         stran(nsecv),dstran(nsecv),predef(*),dpred(*),
     2         props(*),jprops(*),coords(3),time(2),dfgrd(3,3),
     3         curv(2,2),basis(3,3),tss(*)
      dimension FA11(111),FA12(111),FA21(111),FA22(111),FA33(111)
      dimension MB11(111),MB12(111),MB21(111),MB22(111),MB33(111)
      dimension FC11(111),FC12(111),FC21(111),FC22(111),FC33(111)
      dimension MD11(111),MD12(111),MD21(111),MD22(111),MD33(111)
      dimension eA11(111),eA12(111),eA21(111),eA22(111),eA33(111)
      dimension eB11(111),eB12(111),eB21(111),eB22(111),eB33(111)
      dimension eC11(111),eC12(111),eC21(111),eC22(111),eC33(111)
      dimension eD11(111),eD12(111),eD21(111),eD22(111),eD33(111)
      dimension STIF(110,6,6),e(110,6,6)
      integer i,ii,j,jj,kk,m,k,n,curr_ID, next_ID,ij
      double precision e_int1,e_int2,stranCurrent,stranNext
      double precision A11nonl,A22nonl,k_nl
      parameter (zero=0.0d0,half=0.5d0)

C STRESS RESULTANT vs. STRAIN RELATIONS / RVE ANALYSIS --------------------------------

C 1.PRE-DEFINED STRAIN INTERVALS

      eA11=(/-2.000e+00,-1.000e+00,-5.000e-01,-2.950e-01,-2.850e-01,
     + -2.750e-01,-2.650e-01,-2.550e-01,-2.450e-01,-2.350e-01,
     + -2.250e-01,-2.150e-01,-2.050e-01,-1.950e-01,-1.850e-01,
     + -1.750e-01,-1.650e-01,-1.550e-01,-1.450e-01,-1.350e-01,
     + -1.250e-01,-1.150e-01,-1.050e-01,-9.500e-02,-8.500e-02,
     + -7.500e-02,-6.500e-02,-5.500e-02,-4.500e-02,-3.500e-02,
     + -2.500e-02,-1.500e-02,-1.400e-02,-1.300e-02,-1.200e-02,
     + -1.100e-02,-1.000e-02,-9.000e-03,-8.000e-03,-7.000e-03,
     + -6.000e-03,-5.500e-03,-5.000e-03,-4.500e-03,-4.000e-03,
     + -3.500e-03,-3.000e-03,-2.500e-03,-2.000e-03,-1.500e-03,
     + -1.000e-03,-5.000e-04,0.000e+00,5.000e-04,1.000e-03,
     + 1.400e-03,1.450e-03,1.475e-03,1.500e-03,1.525e-03,
     + 1.550e-03,1.600e-03,2.000e-03,2.500e-03,3.000e-03,
     + 3.500e-03,4.000e-03,4.500e-03,5.000e-03,5.500e-03,
     + 6.000e-03,7.000e-03,8.000e-03,9.000e-03,1.000e-02,
     + 1.100e-02,1.200e-02,1.300e-02,1.400e-02,1.500e-02,
     + 2.500e-02,3.500e-02,4.500e-02,5.500e-02,6.500e-02,
     + 7.500e-02,8.500e-02,9.500e-02,1.050e-01,1.150e-01,
     + 1.250e-01,1.350e-01,1.450e-01,1.550e-01,1.650e-01,
     + 1.750e-01,1.850e-01,1.950e-01,2.050e-01,2.150e-01,
     + 2.250e-01,2.350e-01,2.450e-01,2.550e-01,2.650e-01,
     + 2.750e-01,2.850e-01,2.950e-01,5.000e-01,1.000e+00,
     + 2.000e+00/)
```

```
      eA12=eA11; eA21=eA11; eA22=eA11; eA33=eA11; eD11=eA11; eD12=eA11;
      eD21=eA11; eD22=eA11; eD33=eA11; eC11=eA11; eC12=eA11; eC21=eA11;
      eC22=eA11; eC33=eA11

C 2. CORRESPONDING FORCES

      FA11=(/-1.150e+09,-5.762e+08,-2.893e+08,-1.716e+08,-1.659e+08,
     + -1.602e+08,-1.544e+08,-1.487e+08,-1.429e+08,-1.372e+08,
     + -1.315e+08,-1.257e+08,-1.200e+08,-1.143e+08,-1.085e+08,
     + -1.028e+08,-9.704e+07,-9.131e+07,-8.557e+07,-7.983e+07,
     + -7.409e+07,-6.835e+07,-6.262e+07,-5.688e+07,-5.114e+07,
     + -4.540e+07,-3.966e+07,-3.392e+07,-2.819e+07,-2.245e+07,
     + -1.671e+07,-1.097e+07,-1.040e+07,-9.824e+06,-9.250e+06,
     + -8.676e+06,-8.102e+06,-7.522e+06,-6.927e+06,-6.318e+06,
     + -5.691e+06,-5.346e+06,-4.968e+06,-4.589e+06,-4.205e+06,
     + -3.817e+06,-3.423e+06,-3.019e+06,-2.591e+06,-2.077e+06,
     + -1.419e+06,-7.107e+05,0.0000e+00,7.0327e+05,1.4065e+06,
     + 1.9691e+06,2.0395e+06,
     + 2.0641e+06,2.0764e+06,2.0789e+06,2.0798e+06,2.0814e+06,
     + 2.0911e+06,2.1027e+06,2.1142e+06,2.1253e+06,2.1363e+06,
     + 2.1470e+06,2.1575e+06,2.1678e+06,2.1779e+06,2.1975e+06,
     + 2.2164e+06,2.2346e+06,2.2522e+06,2.2692e+06,2.2856e+06,
     + 2.3015e+06,2.3170e+06,2.3319e+06,2.4605e+06,2.5610e+06,
     + 2.6421e+06,2.7090e+06,2.7651e+06,2.8126e+06,2.8530e+06,
     + 2.8877e+06,2.9174e+06,2.9430e+06,2.9650e+06,2.9839e+06,
     + 3.0001e+06,3.0138e+06,3.0254e+06,3.0351e+06,3.0431e+06,
     + 3.0496e+06,3.0546e+06,3.0585e+06,3.0612e+06,3.0629e+06,
     + 3.0636e+06,3.0636e+06,3.0627e+06,3.0611e+06,3.0589e+06,
     + 3.0560e+06,2.9225e+06,2.4819e+06,1.8434e+06/)

      FA21=(/1.975e+08,9.845e+07,4.892e+07,2.862e+07,2.763e+07,
     + 2.664e+07,2.564e+07,2.465e+07,2.366e+07,2.267e+07,
     + 2.168e+07,2.069e+07,1.970e+07,1.871e+07,1.772e+07,
     + 1.673e+07,1.574e+07,1.475e+07,1.376e+07,1.277e+07,
     + 1.178e+07,1.079e+07,9.796e+06,8.806e+06,7.815e+06,
     + 6.825e+06,5.834e+06,4.844e+06,3.853e+06,2.863e+06,
     + 1.872e+06,8.815e+05,7.825e+05,6.834e+05,5.844e+05,
     + 4.853e+05,3.862e+05,2.902e+05,2.023e+05,1.232e+05,
     + 5.410e+04,-1.323e+04,-9.075e+04,-1.665e+05,-2.403e+05,
     + -3.116e+05,-3.792e+05,-4.406e+05,-4.859e+05,-4.674e+05,
     + -3.319e+05,-1.680e+05,0.000e+00,1.680e+05,3.319e+05,
     + 4.447e+05,4.574e+05,4.632e+05,4.674e+05,4.717e+05,
     + 4.751e+05,4.813e+05,4.859e+05,4.406e+05,3.792e+05,
     + 3.116e+05,2.403e+05,1.665e+05,9.075e+04,1.323e+04,
     + -5.410e+04,-1.232e+05,-2.023e+05,-2.902e+05,-3.862e+05,
     + -4.853e+05,-5.844e+05,-6.834e+05,-7.825e+05,-8.815e+05,
     + -1.872e+06,-2.863e+06,-3.853e+06,-4.844e+06,-5.834e+06,
     + -6.825e+06,-7.815e+06,-8.806e+06,-9.796e+06,-1.079e+07,
     + -1.178e+07,-1.277e+07,-1.376e+07,-1.475e+07,-1.574e+07,
     + -1.673e+07,-1.772e+07,-1.871e+07,-1.970e+07,-2.069e+07,
     + -2.168e+07,-2.267e+07,-2.366e+07,-2.465e+07,-2.564e+07,
     + -2.664e+07,-2.763e+07,-2.862e+07,-4.892e+07,-9.845e+07,
     + -1.975e+08/)

      FA12=(/2.458e+05,-5.771e+04,-2.095e+05,-2.717e+05,-2.747e+05,
     + -2.778e+05,-2.808e+05,-2.838e+05,-2.869e+05,-2.899e+05,
     + -2.929e+05,-2.960e+05,-2.990e+05,-3.020e+05,-3.051e+05,
     + -3.081e+05,-3.111e+05,-3.142e+05,-3.172e+05,-3.202e+05,
     + -3.233e+05,-3.263e+05,-3.294e+05,-3.324e+05,-3.354e+05,
     + -3.385e+05,-3.415e+05,-3.445e+05,-3.476e+05,-3.506e+05,
     + -3.536e+05,-3.567e+05,-3.570e+05,-3.573e+05,-3.576e+05,
     + -3.579e+05,-3.581e+05,-3.585e+05,-3.588e+05,-3.591e+05,
     + -3.593e+05,-3.594e+05,-3.596e+05,-3.597e+05,-3.598e+05,
     + -3.599e+05,-3.599e+05,-3.598e+05,-3.595e+05,-3.587e+05,
     + -3.329e+05,-1.680e+05,0.000e+00,1.680e+05,3.329e+05,
     + 3.584e+05,3.586e+05,3.587e+05,3.587e+05,3.588e+05,
     + 3.588e+05,3.590e+05,3.595e+05,3.598e+05,3.599e+05,
     + 3.599e+05,3.598e+05,3.597e+05,3.596e+05,3.594e+05,
```

```
+ 3.593e+05,3.591e+05,3.588e+05,3.585e+05,3.581e+05,
+ 3.579e+05,3.576e+05,3.573e+05,3.570e+05,3.567e+05,
+ 3.536e+05,3.506e+05,3.476e+05,3.445e+05,3.415e+05,
+ 3.385e+05,3.354e+05,3.324e+05,3.294e+05,3.263e+05,
+ 3.233e+05,3.202e+05,3.172e+05,3.142e+05,3.111e+05,
+ 3.081e+05,3.051e+05,3.020e+05,2.990e+05,2.960e+05,
+ 2.929e+05,2.899e+05,2.869e+05,2.838e+05,2.808e+05,
+ 2.778e+05,2.747e+05,2.717e+05,2.095e+05,5.771e+04,
+ -2.458e+05/)

 FA22=(/-2.252e+06,-1.725e+06,-1.461e+06,-1.353e+06,-1.348e+06,
+ -1.343e+06,-1.337e+06,-1.332e+06,-1.327e+06,-1.321e+06,
+ -1.316e+06,-1.311e+06,-1.306e+06,-1.300e+06,-1.295e+06,
+ -1.290e+06,-1.285e+06,-1.279e+06,-1.274e+06,-1.269e+06,
+ -1.263e+06,-1.258e+06,-1.253e+06,-1.248e+06,-1.242e+06,
+ -1.237e+06,-1.232e+06,-1.227e+06,-1.221e+06,-1.216e+06,
+ -1.211e+06,-1.205e+06,-1.205e+06,-1.204e+06,-1.204e+06,
+ -1.203e+06,-1.203e+06,-1.202e+06,-1.202e+06,-1.201e+06,
+ -1.201e+06,-1.201e+06,-1.200e+06,-1.200e+06,-1.200e+06,
+ -1.200e+06,-1.200e+06,-1.200e+06,-1.201e+06,-1.202e+06,
+ -1.125e+06,-5.650e+05,0.0000e+00,5.6594e+05,1.1319e+06,
+ 1.5846e+06,1.6412e+06,
+ 1.6483e+06,1.6497e+06,1.6508e+06,1.6517e+06,1.6532e+06,
+ 1.6617e+06,1.6709e+06,1.6800e+06,1.6888e+06,1.6974e+06,
+ 1.7059e+06,1.7142e+06,1.7224e+06,1.7303e+06,1.7458e+06,
+ 1.7608e+06,1.7752e+06,1.7891e+06,1.8025e+06,1.8155e+06,
+ 1.8281e+06,1.8403e+06,1.8522e+06,1.9540e+06,2.0336e+06,
+ 2.0978e+06,2.1508e+06,2.1953e+06,2.2329e+06,2.2649e+06,
+ 2.2924e+06,2.3160e+06,2.3363e+06,2.3537e+06,2.3687e+06,
+ 2.3815e+06,2.3924e+06,2.4016e+06,2.4092e+06,2.4156e+06,
+ 2.4207e+06,2.4247e+06,2.4278e+06,2.4299e+06,2.4312e+06,
+ 2.4318e+06,2.4318e+06,2.4311e+06,2.4298e+06,2.4280e+06,
+ 2.4258e+06,2.3197e+06,1.9700e+06,1.4632e+06/)

 FA33=(/-4.803e+08,-2.411e+08,-1.215e+08,-7.248e+07,-7.009e+07,
+ -6.770e+07,-6.531e+07,-6.291e+07,-6.052e+07,-5.813e+07,
+ -5.574e+07,-5.335e+07,-5.095e+07,-4.856e+07,-4.617e+07,
+ -4.378e+07,-4.139e+07,-3.899e+07,-3.660e+07,-3.421e+07,
+ -3.182e+07,-2.943e+07,-2.704e+07,-2.464e+07,-2.225e+07,
+ -1.986e+07,-1.747e+07,-1.508e+07,-1.268e+07,-1.029e+07,
+ -7.900e+06,-5.355e+06,-5.077e+06,-4.796e+06,-4.505e+06,
+ -4.202e+06,-3.878e+06,-3.535e+06,-3.165e+06,-2.773e+06,
+ -2.378e+06,-2.180e+06,-1.982e+06,-1.784e+06,-1.586e+06,
+ -1.387e+06,-1.189e+06,-9.911e+05,-7.928e+05,-5.947e+05,
+ -3.965e+05,-1.982e+05,0.000e+00,1.982e+05,3.965e+05,
+ 5.551e+05,5.749e+05,5.848e+05,5.947e+05,6.046e+05,
+ 6.145e+05,6.343e+05,7.928e+05,9.911e+05,1.189e+06,
+ 1.387e+06,1.586e+06,1.784e+06,1.982e+06,2.180e+06,
+ 2.378e+06,2.773e+06,3.165e+06,3.535e+06,3.878e+06,
+ 4.202e+06,4.505e+06,4.796e+06,5.077e+06,5.355e+06,
+ 7.900e+06,1.029e+07,1.268e+07,1.508e+07,1.747e+07,
+ 1.986e+07,2.225e+07,2.464e+07,2.704e+07,2.943e+07,
+ 3.182e+07,3.421e+07,3.660e+07,3.899e+07,4.139e+07,
+ 4.378e+07,4.617e+07,4.856e+07,5.095e+07,5.335e+07,
+ 5.574e+07,5.813e+07,6.052e+07,6.291e+07,6.531e+07,
+ 6.770e+07,7.009e+07,7.248e+07,1.215e+08,2.411e+08,
+ 4.803e+08/)

 FC11=(/1.392e+07,6.629e+06,2.983e+06,1.488e+06,1.416e+06,
+ 1.343e+06,1.270e+06,1.197e+06,1.125e+06,1.053e+06,
+ 9.803e+05,9.094e+05,8.381e+05,7.677e+05,6.976e+05,
+ 6.281e+05,5.590e+05,4.903e+05,4.229e+05,3.554e+05,
+ 2.895e+05,2.243e+05,1.619e+05,1.059e+05,6.656e+04,
+ 4.011e+04,2.269e+04,1.238e+04,6.614e+03,3.380e+03,
+ 1.531e+03,5.164e+02,4.582e+02,4.000e+02,3.418e+02,
+ 2.836e+02,2.253e+02,1.828e+02,1.506e+02,1.184e+02,
+ 8.627e+01,7.019e+01,5.438e+01,4.894e+01,4.350e+01,
+ 3.807e+01,3.263e+01,2.719e+01,2.175e+01,1.631e+01,
+ 1.088e+01,5.438e+00,0.000e+00,-5.438e+00,-1.088e+01,
```

```
     + -1.523e+01,-1.577e+01,-1.604e+01,-1.631e+01,-1.659e+01,
     + -1.686e+01,-1.740e+01,-2.175e+01,-2.719e+01,-3.263e+01,
     + -3.807e+01,-4.350e+01,-4.894e+01,-5.438e+01,-7.019e+01,
     + -8.627e+01,-1.184e+02,-1.506e+02,-1.828e+02,-2.253e+02,
     + -2.836e+02,-3.418e+02,-4.000e+02,-4.582e+02,-5.164e+02,
     + -1.531e+03,-3.380e+03,-6.614e+03,-1.238e+04,-2.269e+04,
     + -4.011e+04,-6.656e+04,-1.059e+05,-1.619e+05,-2.243e+05,
     + -2.895e+05,-3.554e+05,-4.229e+05,-4.903e+05,-5.590e+05,
     + -6.281e+05,-6.976e+05,-7.677e+05,-8.381e+05,-9.094e+05,
     + -9.803e+05,-1.053e+06,-1.125e+06,-1.197e+06,-1.270e+06,
     + -1.343e+06,-1.416e+06,-1.488e+06,-2.983e+06,-6.629e+06,
     + -1.392e+07/)

      FC12=(/7.101e+06,3.462e+06,1.642e+06,8.961e+05,8.594e+05,
     + 8.230e+05,7.865e+05,7.500e+05,7.136e+05,6.769e+05,
     + 6.403e+05,6.040e+05,5.675e+05,5.310e+05,4.943e+05,
     + 4.579e+05,4.214e+05,3.846e+05,3.484e+05,3.117e+05,
     + 2.754e+05,2.392e+05,2.031e+05,1.669e+05,1.312e+05,
     + 9.626e+04,6.267e+04,3.257e+04,1.179e+04,3.658e+03,
     + 1.242e+03,3.650e+02,3.040e+02,2.491e+02,2.183e+02,
     + 1.875e+02,1.567e+02,1.259e+02,9.513e+01,6.930e+01,
     + 5.940e+01,5.445e+01,4.950e+01,4.455e+01,3.960e+01,
     + 3.465e+01,2.970e+01,2.475e+01,1.980e+01,1.485e+01,
     + 9.901e+00,4.950e+00,0.000e+00,-4.950e+00,-9.901e+00,
     + -1.386e+01,-1.436e+01,-1.460e+01,-1.485e+01,-1.510e+01,
     + -1.535e+01,-1.584e+01,-1.980e+01,-2.475e+01,-2.970e+01,
     + -3.465e+01,-3.960e+01,-4.455e+01,-4.950e+01,-5.445e+01,
     + -5.940e+01,-6.930e+01,-9.513e+01,-1.259e+02,-1.567e+02,
     + -1.875e+02,-2.183e+02,-2.491e+02,-3.040e+02,-3.650e+02,
     + -1.242e+03,-3.658e+03,-1.179e+04,-3.257e+04,-6.267e+04,
     + -9.626e+04,-1.312e+05,-1.669e+05,-2.031e+05,-2.392e+05,
     + -2.754e+05,-3.117e+05,-3.484e+05,-3.846e+05,-4.214e+05,
     + -4.579e+05,-4.943e+05,-5.310e+05,-5.675e+05,-6.040e+05,
     + -6.403e+05,-6.769e+05,-7.136e+05,-7.500e+05,-7.865e+05,
     + -8.230e+05,-8.594e+05,-8.961e+05,-1.642e+06,-3.462e+06,
     + -7.101e+06/)

      FC21=(/9.907e+06,4.730e+06,2.142e+06,1.081e+06,1.029e+06,
     + 9.774e+05,9.257e+05,8.738e+05,8.221e+05,7.703e+05,
     + 7.187e+05,6.675e+05,6.158e+05,5.647e+05,5.135e+05,
     + 4.625e+05,4.118e+05,3.610e+05,3.110e+05,2.608e+05,
     + 2.116e+05,1.625e+05,1.154e+05,7.303e+04,4.423e+04,
     + 2.585e+04,1.415e+04,7.465e+03,3.872e+03,1.935e+03,
     + 8.635e+02,2.886e+02,2.560e+02,2.234e+02,1.908e+02,
     + 1.582e+02,1.256e+02,1.018e+02,8.383e+01,6.586e+01,
     + 4.790e+01,3.891e+01,3.008e+01,2.707e+01,2.407e+01,
     + 2.106e+01,1.805e+01,1.504e+01,1.203e+01,9.025e+00,
     + 6.016e+00,3.008e+00,0.000e+00,-3.008e+00,-6.016e+00,
     + -8.423e+00,-8.724e+00,-8.874e+00,-9.025e+00,-9.175e+00,
     + -9.325e+00,-9.626e+00,-1.203e+01,-1.504e+01,-1.805e+01,
     + -2.106e+01,-2.407e+01,-2.707e+01,-3.008e+01,-3.891e+01,
     + -4.790e+01,-6.586e+01,-8.383e+01,-1.018e+02,-1.256e+02,
     + -1.582e+02,-1.908e+02,-2.234e+02,-2.560e+02,-2.886e+02,
     + -8.635e+02,-1.935e+03,-3.872e+03,-7.465e+03,-1.415e+04,
     + -2.585e+04,-4.423e+04,-7.303e+04,-1.154e+05,-1.625e+05,
     + -2.116e+05,-2.608e+05,-3.110e+05,-3.610e+05,-4.118e+05,
     + -4.625e+05,-5.135e+05,-5.647e+05,-6.158e+05,-6.675e+05,
     + -7.187e+05,-7.703e+05,-8.221e+05,-8.738e+05,-9.257e+05,
     + -9.774e+05,-1.029e+06,-1.081e+06,-2.142e+06,-4.730e+06,
     + -9.907e+06/)

      FC22=(/2.330e+07,1.138e+07,5.416e+06,2.968e+06,2.847e+06,
     + 2.728e+06,2.607e+06,2.487e+06,2.367e+06,2.246e+06,
     + 2.125e+06,2.005e+06,1.885e+06,1.764e+06,1.642e+06,
     + 1.522e+06,1.401e+06,1.280e+06,1.159e+06,1.038e+06,
     + 9.176e+05,7.970e+05,6.770e+05,5.564e+05,4.374e+05,
     + 3.203e+05,2.074e+05,1.058e+05,3.582e+04,9.618e+03,
     + 2.786e+03,7.208e+02,5.892e+02,4.721e+02,4.128e+02,
     + 3.536e+02,2.943e+02,2.350e+02,1.757e+02,1.262e+02,
```

```
+ 1.082e+02,9.916e+01,9.015e+01,8.113e+01,7.212e+01,
+ 6.310e+01,5.409e+01,4.507e+01,3.606e+01,2.704e+01,
+ 1.803e+01,9.015e+00,0.000e+00,-9.015e+00,-1.803e+01,
+ -2.524e+01,-2.614e+01,-2.659e+01,-2.704e+01,-2.750e+01,
+ -2.795e+01,-2.885e+01,-3.606e+01,-4.507e+01,-5.409e+01,
+ -6.310e+01,-7.212e+01,-8.113e+01,-9.015e+01,-9.916e+01,
+ -1.082e+02,-1.262e+02,-1.757e+02,-2.350e+02,-2.943e+02,
+ -3.536e+02,-4.128e+02,-4.721e+02,-5.892e+02,-7.208e+02,
+ -2.786e+03,-9.618e+03,-3.582e+04,-1.058e+05,-2.074e+05,
+ -3.203e+05,-4.374e+05,-5.564e+05,-6.770e+05,-7.970e+05,
+ -9.176e+05,-1.038e+06,-1.159e+06,-1.280e+06,-1.401e+06,
+ -1.522e+06,-1.642e+06,-1.764e+06,-1.885e+06,-2.005e+06,
+ -2.125e+06,-2.246e+06,-2.367e+06,-2.487e+06,-2.607e+06,
+ -2.728e+06,-2.847e+06,-2.968e+06,-5.416e+06,-1.138e+07,
+ -2.330e+07/)

 MD11=(/-7.394e+05,-3.797e+05,-1.999e+05,-1.262e+05,-1.226e+05,
+ -1.190e+05,-1.154e+05,-1.117e+05,-1.081e+05,-1.044e+05,
+ -1.008e+05,-9.712e+04,-9.340e+04,-8.967e+04,-8.591e+04,
+ -8.213e+04,-7.832e+04,-7.446e+04,-7.061e+04,-6.669e+04,
+ -6.276e+04,-5.875e+04,-5.470e+04,-5.045e+04,-4.580e+04,
+ -4.082e+04,-3.563e+04,-3.029e+04,-2.484e+04,-1.935e+04,
+ -1.383e+04,-8.302e+03,-7.749e+03,-7.196e+03,-6.643e+03,
+ -6.090e+03,-5.537e+03,-4.983e+03,-4.429e+03,-3.875e+03,
+ -3.322e+03,-3.045e+03,-2.768e+03,-2.491e+03,-2.214e+03,
+ -1.937e+03,-1.661e+03,-1.384e+03,-1.107e+03,-8.303e+02,
+ -5.536e+02,-2.768e+02,-4.547e-13,2.768e+02,5.536e+02,
+ 7.750e+02,8.027e+02,8.165e+02,8.303e+02,8.442e+02,
+ 8.580e+02,8.857e+02,1.107e+03,1.384e+03,1.661e+03,
+ 1.937e+03,2.214e+03,2.491e+03,2.768e+03,3.045e+03,
+ 3.322e+03,3.875e+03,4.429e+03,4.983e+03,5.537e+03,
+ 6.090e+03,6.643e+03,7.196e+03,7.749e+03,8.302e+03,
+ 1.383e+04,1.935e+04,2.484e+04,3.029e+04,3.563e+04,
+ 4.082e+04,4.580e+04,5.045e+04,5.470e+04,5.875e+04,
+ 6.276e+04,6.669e+04,7.061e+04,7.446e+04,7.832e+04,
+ 8.213e+04,8.591e+04,8.967e+04,9.340e+04,9.712e+04,
+ 1.008e+05,1.044e+05,1.081e+05,1.117e+05,1.154e+05,
+ 1.190e+05,1.226e+05,1.262e+05,1.999e+05,3.797e+05,
+ 7.394e+05/)

 MD12=(/-9.593e+04,-5.238e+04,-3.061e+04,-2.149e+04,-2.101e+04,
+ -2.053e+04,-2.004e+04,-1.955e+04,-1.905e+04,-1.854e+04,
+ -1.802e+04,-1.751e+04,-1.698e+04,-1.645e+04,-1.591e+04,
+ -1.537e+04,-1.481e+04,-1.424e+04,-1.367e+04,-1.309e+04,
+ -1.249e+04,-1.189e+04,-1.127e+04,-1.063e+04,-9.967e+03,
+ -9.279e+03,-8.539e+03,-7.700e+03,-6.631e+03,-5.275e+03,
+ -3.794e+03,-2.282e+03,-2.130e+03,-1.979e+03,-1.826e+03,
+ -1.674e+03,-1.522e+03,-1.370e+03,-1.218e+03,-1.066e+03,
+ -9.135e+02,-8.374e+02,-7.613e+02,-6.851e+02,-6.090e+02,
+ -5.329e+02,-4.568e+02,-3.806e+02,-3.045e+02,-2.284e+02,
+ -1.523e+02,-7.613e+01,0.000e+00,7.613e+01,1.523e+02,
+ 2.132e+02,2.208e+02,2.246e+02,2.284e+02,2.322e+02,
+ 2.360e+02,2.436e+02,3.045e+02,3.806e+02,4.568e+02,
+ 5.329e+02,6.090e+02,6.851e+02,7.613e+02,8.374e+02,
+ 9.135e+02,1.066e+03,1.218e+03,1.370e+03,1.522e+03,
+ 1.674e+03,1.826e+03,1.979e+03,2.130e+03,2.282e+03,
+ 3.794e+03,5.275e+03,6.631e+03,7.700e+03,8.539e+03,
+ 9.279e+03,9.967e+03,1.063e+04,1.127e+04,1.189e+04,
+ 1.249e+04,1.309e+04,1.367e+04,1.424e+04,1.481e+04,
+ 1.537e+04,1.591e+04,1.645e+04,1.698e+04,1.751e+04,
+ 1.802e+04,1.854e+04,1.905e+04,1.955e+04,2.004e+04,
+ 2.053e+04,2.101e+04,2.149e+04,3.061e+04,5.238e+04,
+ 9.593e+04/)

 MD21=(/-4.463e+04,-2.899e+04,-2.118e+04,-1.797e+04,-1.781e+04,
+ -1.766e+04,-1.750e+04,-1.734e+04,-1.718e+04,-1.701e+04,
+ -1.682e+04,-1.663e+04,-1.642e+04,-1.620e+04,-1.597e+04,
+ -1.573e+04,-1.547e+04,-1.519e+04,-1.490e+04,-1.458e+04,
+ -1.425e+04,-1.388e+04,-1.347e+04,-1.292e+04,-1.205e+04,
```

```
     + -1.093e+04,-9.651e+03,-8.262e+03,-6.804e+03,-5.311e+03,
     + -3.801e+03,-2.283e+03,-2.131e+03,-1.979e+03,-1.827e+03,
     + -1.675e+03,-1.523e+03,-1.370e+03,-1.218e+03,-1.066e+03,
     + -9.137e+02,-8.376e+02,-7.614e+02,-6.853e+02,-6.091e+02,
     + -5.330e+02,-4.569e+02,-3.807e+02,-3.046e+02,-2.284e+02,
     + -1.523e+02,-7.614e+01,0.000e+00,7.614e+01,1.523e+02,
     + 2.132e+02,2.208e+02,2.246e+02,2.284e+02,2.322e+02,
     + 2.360e+02,2.437e+02,3.046e+02,3.807e+02,4.569e+02,
     + 5.330e+02,6.091e+02,6.853e+02,7.614e+02,8.376e+02,
     + 9.137e+02,1.066e+03,1.218e+03,1.370e+03,1.523e+03,
     + 1.675e+03,1.827e+03,1.979e+03,2.131e+03,2.283e+03,
     + 3.801e+03,5.311e+03,6.804e+03,8.262e+03,9.651e+03,
     + 1.093e+04,1.205e+04,1.292e+04,1.347e+04,1.388e+04,
     + 1.425e+04,1.458e+04,1.490e+04,1.519e+04,1.547e+04,
     + 1.573e+04,1.597e+04,1.620e+04,1.642e+04,1.663e+04,
     + 1.682e+04,1.701e+04,1.718e+04,1.734e+04,1.750e+04,
     + 1.766e+04,1.781e+04,1.797e+04,2.118e+04,2.899e+04,
     + 4.463e+04/)

      MD22=(/-3.105e+05,-1.711e+05,-1.014e+05,-7.213e+04,-7.057e+04,
     + -6.900e+04,-6.739e+04,-6.577e+04,-6.415e+04,-6.248e+04,
     + -6.079e+04,-5.908e+04,-5.736e+04,-5.560e+04,-5.381e+04,
     + -5.200e+04,-5.015e+04,-4.825e+04,-4.635e+04,-4.439e+04,
     + -4.240e+04,-4.036e+04,-3.827e+04,-3.609e+04,-3.386e+04,
     + -3.150e+04,-2.897e+04,-2.607e+04,-2.238e+04,-1.775e+04,
     + -1.274e+04,-7.650e+03,-7.141e+03,-6.631e+03,-6.121e+03,
     + -5.611e+03,-5.101e+03,-4.591e+03,-4.082e+03,-3.572e+03,
     + -3.061e+03,-2.806e+03,-2.551e+03,-2.296e+03,-2.041e+03,
     + -1.786e+03,-1.531e+03,-1.276e+03,-1.020e+03,-7.653e+02,
     + -5.102e+02,-2.551e+02,0.000e+00,2.551e+02,5.102e+02,
     + 7.143e+02,7.398e+02,7.526e+02,7.653e+02,7.781e+02,
     + 7.908e+02,8.164e+02,1.020e+03,1.276e+03,1.531e+03,
     + 1.786e+03,2.041e+03,2.296e+03,2.551e+03,2.806e+03,
     + 3.061e+03,3.572e+03,4.082e+03,4.591e+03,5.101e+03,
     + 5.611e+03,6.121e+03,6.631e+03,7.141e+03,7.650e+03,
     + 1.274e+04,1.775e+04,2.238e+04,2.607e+04,2.897e+04,
     + 3.150e+04,3.386e+04,3.609e+04,3.827e+04,4.036e+04,
     + 4.240e+04,4.439e+04,4.635e+04,4.825e+04,5.015e+04,
     + 5.200e+04,5.381e+04,5.560e+04,5.736e+04,5.908e+04,
     + 6.079e+04,6.248e+04,6.415e+04,6.577e+04,6.739e+04,
     + 6.900e+04,7.057e+04,7.213e+04,1.014e+05,1.711e+05,
     + 3.105e+05/)

      MD33=(/-2.687e+05,-1.546e+05,-9.021e+04,-5.649e+04,-5.460e+04,
     + -5.272e+04,-5.083e+04,-4.894e+04,-4.704e+04,-4.514e+04,
     + -4.324e+04,-4.133e+04,-3.942e+04,-3.751e+04,-3.560e+04,
     + -3.368e+04,-3.176e+04,-2.984e+04,-2.792e+04,-2.600e+04,
     + -2.407e+04,-2.215e+04,-2.023e+04,-1.831e+04,-1.638e+04,
     + -1.446e+04,-1.253e+04,-1.060e+04,-8.677e+03,-6.748e+03,
     + -4.820e+03,-2.892e+03,-2.699e+03,-2.507e+03,-2.314e+03,
     + -2.121e+03,-1.928e+03,-1.735e+03,-1.542e+03,-1.350e+03,
     + -1.157e+03,-1.060e+03,-9.640e+02,-8.676e+02,-7.712e+02,
     + -6.748e+02,-5.784e+02,-4.820e+02,-3.856e+02,-2.892e+02,
     + -1.928e+02,-9.640e+01,0.000e+00,9.640e+01,1.928e+02,
     + 2.699e+02,2.796e+02,2.844e+02,2.892e+02,2.940e+02,
     + 2.989e+02,3.085e+02,3.856e+02,4.820e+02,5.784e+02,
     + 6.748e+02,7.712e+02,8.676e+02,9.640e+02,1.060e+03,
     + 1.157e+03,1.350e+03,1.542e+03,1.735e+03,1.928e+03,
     + 2.121e+03,2.314e+03,2.507e+03,2.699e+03,2.892e+03,
     + 4.820e+03,6.748e+03,8.677e+03,1.060e+04,1.253e+04,
     + 1.446e+04,1.638e+04,1.831e+04,2.023e+04,2.215e+04,
     + 2.407e+04,2.600e+04,2.792e+04,2.984e+04,3.176e+04,
     + 3.368e+04,3.560e+04,3.751e+04,3.942e+04,4.133e+04,
     + 4.324e+04,4.514e+04,4.704e+04,4.894e+04,5.083e+04,
     + 5.272e+04,5.460e+04,5.649e+04,9.021e+04,1.546e+05,
     + 2.687e+05/)

C -----------------------------------------------------------------------------------------------------------
```

```
C STATE VARIABLE DEFINITION

      statev(1)=noel; statev(2)=npt; statev(3)=kstep; statev(4)=kinc;
      statev(5)=kit; statev(6)=time(1); statev(7)=dtime;statev(8)=thick;
      statev(9)=celent; statev(10)=temp; statev(11)=dtemp;
      statev(12)=zero; statev(13)=zero
      k3=0
      do k1=1,3
        do k2=1,3
          statev(14+k3)=basis(k1,k2)
          statev(23+k3)=dfgrd(k1,k2)
          k3=k3+1
        end do
      end do
      statev(32)=curv(1,1); statev(33)=curv(1,2); statev(34)=curv(2,1)
      statev(35)=curv(2,2); statev(36)=tss(1); statev(37)=tss(2)
      statev(38)=coords(1); statev(39)=coords(2); statev(40)=coords(3)
      do k1=1,nsecv
        statev(40+k1)=stran(k1)
      end do
      istatev=40+nsecv

C JACOBIAN DEFINITION

        ddndde(1,1) = props(1)
        ddndde(1,2) = props(2)
        ddndde(2,2) = props(3)
        ddndde(1,3) = props(4)
        ddndde(2,3) = props(5)
        ddndde(3,3) = props(6)
        ddndde(1,4) = props(7)
        ddndde(2,4) = props(8)
        ddndde(3,4) = props(9)
        ddndde(4,4) = props(10)
        ddndde(1,5) = props(11)
        ddndde(2,5) = props(12)
        ddndde(3,5) = props(13)
        ddndde(4,5) = props(14)
        ddndde(5,5) = props(15)
        ddndde(1,6) = props(16)
        ddndde(2,6) = props(17)
        ddndde(3,6) = props(18)
        ddndde(4,6) = props(19)
        ddndde(5,6) = props(20)
        ddndde(6,6) = props(21)
        do i=1,nsecv
          do j=1,i
            ddndde(i,j) = ddndde(j,i)
          end do
        end do

C INITIALIZING STIFFNESS AND STRAIN ARRAYS

      do ii=1,110
        do jj=1,6
          do kk=1,6
            STIF(ii,jj,kk)= 0.0e-15
            e(ii,jj,kk)= 1.0e-16
          end do
        end do
      end do

C ASSEMBLE STIFFNESS 3D ARRAY

      do m=2,111
        STIF(m-1,1,1) = (FA11(m)-FA11(m-1))/(eA11(m)-eA11(m-1))
        STIF(m-1,1,2) = (FA12(m)-FA12(m-1))/(eA12(m)-eA12(m-1))
        STIF(m-1,1,4) = (FC11(m)-FC11(m-1))/(eC11(m)-eC11(m-1))
        STIF(m-1,1,5) = (FC12(m)-FC12(m-1))/(eC12(m)-eC12(m-1))
```

```
      STIF(m-1,2,1) = (FA21(m)-FA21(m-1))/(eA21(m)-eA21(m-1))
      STIF(m-1,2,2) = (FA22(m)-FA22(m-1))/(eA22(m)-eA22(m-1))
      STIF(m-1,2,4) = (FC21(m)-FC21(m-1))/(eC21(m)-eC21(m-1))
      STIF(m-1,2,5) = (FC22(m)-FC22(m-1))/(eC22(m)-eC22(m-1))
      STIF(m-1,3,3) = (FA33(m)-FA33(m-1))/(eA33(m)-eA33(m-1))
      STIF(m-1,4,4) = (MD11(m)-MD11(m-1))/(eD11(m)-eD11(m-1))
      STIF(m-1,4,5) = (MD12(m)-MD12(m-1))/(eD12(m)-eD12(m-1))
      STIF(m-1,5,4) = (MD21(m)-MD21(m-1))/(eD21(m)-eD21(m-1))
      STIF(m-1,5,5) = (MD22(m)-MD22(m-1))/(eD22(m)-eD22(m-1))
      STIF(m-1,6,6) = (MD33(m)-MD33(m-1))/(eD33(m)-eD33(m-1))
    end do

C ASSEMBLE STRAIN 3D ARRAY

    do m=2,111
      e(m-1,1,1) = eA11(m)
      e(m-1,2,1) = eA21(m)
      e(m-1,4,1) = eB11(m)
      e(m-1,5,1) = eB21(m)
      e(m-1,1,2) = eA12(m)
      e(m-1,2,2) = eA22(m)
      e(m-1,4,2) = eB12(m)
      e(m-1,5,2) = eB22(m)
      e(m-1,3,3) = eA33(m)
      e(m-1,6,3) = eB33(m)
      e(m-1,1,4) = eC11(m)
      e(m-1,2,4) = eC21(m)
      e(m-1,4,4) = eD11(m)
      e(m-1,5,4) = eD21(m)
      e(m-1,1,5) = eC12(m)
      e(m-1,2,5) = eC22(m)
      e(m-1,4,5) = eD12(m)
      e(m-1,5,5) = eD22(m)
      e(m-1,3,6) = eC33(m)
      e(m-1,6,6) = eD33(m)
    end do

C FORCE/MOMENT UPDATE CODE ---------------------------------------------------------------------

C Loop over force(i) and strain(j)
C Shell stiffness based on ID,i,j, i.e. F(i) = K(ID,i,j)*e(j)

      do i=1,6
      do j=1,6

C CHECK STRAIN ID IN THE ASSEMBLED STRAIN MATRIX

C Check current strain ID
C Start ID = 53 if tension / 52 if compression

      curr_ID=53
      ii=53

11     if (curr_ID.EQ.53) then
          stranCurrent=stran(j)
      if              (stranCurrent .EQ. 0.0) then
          if (dstran(j) .LT. 0.0) then
            curr_ID=52
          end if
      else if (stranCurrent .GT. 0.0) then
         if (e(ii,i,j).GT.stranCurrent) then
            curr_ID=ii
         else
            ii=ii+1
            goto 11
         end if
      else if (stranCurrent .LT. 0.0) then
          if (e((ii-1),i,j).LT.stranCurrent) then
            curr_ID=ii
```

```
          else
            ii=ii-1
            goto 11
          end if
        else
        endif
        endif

C Check next strain ID

        next_ID=53
        jj=53

12      if (next_ID.EQ.53) then
          stranNext=stran(j)+dstran(j)
        if (stranNext .GT. 0.0) then
          if (e(jj,i,j).GT.stranNext) then
            next_ID=jj
          else
            jj=jj+1
            goto 12
          end if
        else if (stranNext .LT. 0.0) then
          if (e((jj-1),i,j).LT.stranNext) then
            next_ID=jj
          else
            jj=jj-1
            goto 12
          end if
        endif
        endif

C Adjusting strain ID if out of array bounds

      if (STIF(53,i,j).EQ.0.0e-15) then
          curr_ID = 53
          next_ID = 53
      end if

C Check strain and increment signs

      curr_stran_modif=sign(stran(j),dstran(j))

C UPDATE FORCES AND MOMENTS

C Check how many intervals to move in the stiffness table

      n=next_ID-curr_ID

C Reduce STIF(ID,i,j) if shell yielding in 11 or 22 directions
      ij=(i-1)*6+j

      if (stran(1).GE.1.478e-3) then
       if (stran(1).GT.stran(2)) then
        if (ij.EQ.1) then
            A11nonl=STIF(curr_ID,i,j)
            k_nl=A11nonl/PROPS(1)
        else
          select case (ij)
            case (2);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
            case (4);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(7)*k_nl
            case (5);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(11)*k_nl
            case (7);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
            case (8);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(3)*k_nl
            case (10);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(8)*k_nl
            case (11);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(12)*k_nl
            case (15);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(6)
            case (22);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(10)*k_nl
            case (23);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
```

```
            case (28);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
            case (29);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(15)*k_nl
            case (36);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(21)
            case default; curr_ID=1;n=0;STIF(curr_ID,i,j)=0.0
          end select
       end if
      end if
     end if

     if (stran(2).GE.1.458e-3) then
      if (stran(2).GT.stran(1)) then
       if (ij.EQ.8) then
            A22nonl=STIF(curr_ID,i,j)
            k_nl=A22nonl/PROPS(3)
       else
          select case (ij)
            case (1);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(1)*k_nl
            case (2);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
            case (4);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(7)*k_nl
            case (5);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(11)*k_nl
            case (7);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(2)*k_nl
            case (10);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(8)*k_nl
            case (11);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(12)*k_nl
            case (15);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(6)
            case (22);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(10)*k_nl
            case (23);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
            case (28);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(14)*k_nl
            case (29);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(15)*k_nl
            case (36);curr_ID=1;n=0;STIF(curr_ID,i,j)=PROPS(21)
            case default; curr_ID=1;n=0;STIF(curr_ID,i,j)=0.0
          end select
       end if
      end if
     end if

C CASE 1: Increment is in the same direction as strain

       if (curr_stran_modif.EQ.stran(j)) then

C n = 0, use stiffness of interval

        if (n.EQ.0) then
       force(i) = force(i)+ STIF(curr_ID,i,j)*dstran(j)

C n =/= 0, interpolate

C Interpolation: tension

        else if (n.GT.0) then

          e_int1 = e(curr_ID,i,j) - stranCurrent
          e_int2 = stranNext - e((curr_ID+n-1),i,j)

          force(i) = force(i) + STIF(curr_ID,i,j)*e_int1
          force(i) = force(i) + STIF((curr_ID+n),i,j)*e_int2

          do k=2,n
            force(i) = force(i) + STIF((curr_ID+k-1),i,j)*
    + (e((curr_ID+k-1),i,j) - e((curr_ID+k-2),i,j))
          end do

C Interpolation: compression

         else

          e_int1 = e((curr_ID-1),i,j) - stranCurrent
          e_int2 = stranNext - e((curr_ID+n),i,j)

          force(i) = force(i) + STIF(curr_ID,i,j)*e_int1
```

```
          force(i) = force(i) + STIF((curr_ID+n),i,j)*e_int2

          do k=-2,n,-1
            force(i) = force(i) + STIF((curr_ID+k+1),i,j)*
     + (e((curr_ID+k),i,j) - e((curr_ID+k+1),i,j))
          end do

        end if

C CASE 2: Increment is in opposite same direction as strain

       else

C n = 0, use stiffness of interval

         if (n.EQ.0) then
        force(i) = force(i) + STIF(curr_ID,i,j)*dstran(j)

C n =/= 0, interpolate

C Interpolation: tension

         else if (n.LT.0) then

           e_int1 = stranCurrent - e((curr_ID-1),i,j)
           e_int2 = e((curr_ID+n),i,j) - stranNext

           force(i) = force(i) - STIF(curr_ID,i,j)*e_int1
           force(i) = force(i) - STIF((curr_ID+n),i,j)*e_int2

           do k=-2,n,-1
             force(i) = force(i) - STIF((curr_ID+k+1),i,j)*
     + (e((curr_ID+k+1),i,j) - e((curr_ID+k),i,j))
           end do

C Interpolation: compression

          else

           e_int1 = stranCurrent - e(curr_ID,i,j)
           e_int2 = e((curr_ID+n-1),i,j)-stranNext

           force(i) = force(i) - STIF(curr_ID,i,j)*e_int1
           force(i) = force(i) - STIF((curr_ID+n),i,j)*e_int2

           do k=2,n
             force(i) = force(i) - STIF((curr_ID+k-1),i,j)*
     + (e((curr_ID+k-2),i,j)-e((curr_ID+k-1),i,j))
           end do

         end if

       end if

C -------------------------------------------------------------------------------------------------------------

        end do
         statev(istatev+i) = force(i)
       end do

 999  continue

      return
      end
```

This report describes an Abaqus subroutine for geometric and material nonlinear analysis of periodic panels using the first-order shear deformation theory. The structure is modelled with shell elements, as one layer of equivalent mechanical properties. The subroutine modifies the stiffness matrix of each element of the mesh separately based on its strain state. It is based on pre-computed stiffness curves that define the ABCD stiffness matrix of a unit cell. By looking at combinations of force and strain, the code interpolates the stiffness curves to calculate equivalent nonlinear stiffness. The code presented and examples deal with web-core sandwich panels, yet it may be readily extended to other periodic structures. The examples show that the subroutine can describe nonlinearities such as global buckling, local buckling and post-yield response with good accuracy and low computational cost compared to conventional 3D FEM analysis.

ISBN 978-952-60-6905-0 (pdf)
ISSN-L 1799-4896
ISSN 1799-4896 (printed)
ISSN 1799-490X (pdf)

Aalto University
School of Engineering
Department of Mechanical Engineering
www.aalto.fi

BUSINESS + ECONOMY

ART + DESIGN + ARCHITECTURE

SCIENCE + TECHNOLOGY

CROSSOVER

DOCTORAL DISSERTATIONS

Aalto-ST 9/2016