TU DELFT

REN SHIBO

ANALYSIS OF TENT STRUCTURES

Implementation of non-linear material behavior in membrane structural analysis

Analysis of Tent Structures

Implementation of non-linear material behavior in membrane structural analysis

Additional Master Thesis

Ren Shibo
Student ID 1330209

Report submitted in fulfillment for an additional Master Thesis Project
in Structural Engineering, at the Delft University of Technology.

Supervisors:
Dr.ir. P.C.J. Hoogenboom
Prof.dr.ir. J.G. Rots

Delft University of Technology
Faculty of Civil Engineering and Geosciences

March 2008

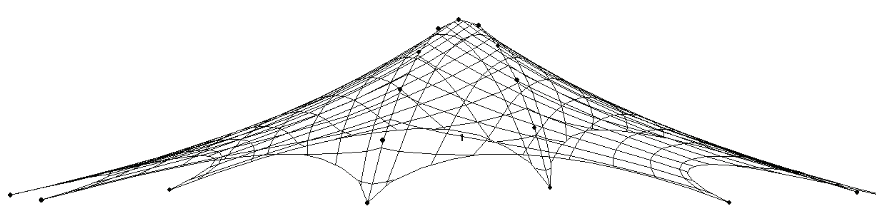


Table of Contents

PREFACE .... **VI**

SUMMARY .... **VII**

1 INTRODUCTION **1**

1.1 GENERAL 1
1.2 PTFE COATED FIBERGLASS FABRIC 2

2 STRUCTURAL MODEL FOR TEXTILE **4**

3 MEMBRANE STRUCTURAL ANALYSIS IN ANSYS **7**

3.1 GEOMETRY 7
3.2 ELEMENT 8
3.3 LOAD AND PRESTRESS 9
3.4 ISOTROPIC MATERIAL MODEL 10
3.5 ISOTROPIC MATERIAL MODEL-LINEAR ANALYSIS 11
3.6 ISOTROPIC MATERIAL MODEL-GEOMETRIC NONLINEAR ANALYSIS 12
3.6.1 Settings for nonlinear analysis 12
3.6.2 Prestress by temperature load 14
3.6.3 Nonlinear analysis under wind load 16
3.6.4 Verification by membrane element-SHELL41 19

4 ANSYS USER MATERIAL SUBROUTINE USERMAT **21**

4.1 SETTING UP A USERMAT 21
4.2 COMPILING AND LINKING USERMAT 23
4.3 VERIFICATION OF USERMAT 24

5 ANALYSIS WITH NONLINEAR USERMAT MODEL **26**

5.1 USERMAT-I 26
5.2 USERMAT-I RESULT 29
5.3 USERMAT-II 32
5.4 USERMAT-II LOAD STEP I PRESTRESS 35
5.5 USERMAT-II LOAD STEP II WIND LOAD 37

6 CONCLUSIONS AND RECOMMENDATIONS **40**

6.1 CONCLUSIONS 40
6.2 RECOMMENDATIONS 41

REFERENCE **43**

APPENDIX-I TIPS FOR USERMAT LINKING **44**

APPENDIX-II USERMAT – LINEAR ISOTROPIC MODEL **46**

APPENDIX-III USERMAT-I AND APDL SCRIPT-I **48**

APPENDIX-IV USERMAT-II AND APDL SCRIPT-II **51**

APPENDIX-V USERMAT - TRANSFER BETWEEN XY AND PRINCIPAL DIRECTION **55**

APPENDIX-VI CD ROM CONTENTS **58**

Preface

This report documents the eight-week additional thesis project which I undertook in Faculty of Civil Engineering and Geosciences at the Delft University of Technology.

This study wouldn't have been successfully completed without the help of Dr.ir. P.C.J. Hoogenboom and therefore I would like to sincerely thank him. I am also indebted to Prof.dr.ir. J.G. Rots for his kindness to be one of the committee members for this project.

Delft, March 2008

Summary

In this additional master thesis, the possibilities of implementing a nonlinear material model in analysis of tent structures are investigated. The structural model for textile is programmed in the FORTRAN language and is linked to the general purpose finite element software ANSYS. Several subroutines have been created and tested in order to improve the applicability. A tent structure has been modeled and analyzed with this nonlinear material model. Prestress is introduced by a temperature increment and subsequently a vertical load is applied. The computed behavior is realistic, both globally and in a selected element. Effort is also spent on convergence matters to make the calculation time acceptable for engineering application.

1 Introduction

1.1 General

Tensioned membrane structures are widely used in current practice. The behavior of such structures is complicated due to the nonlinear structural behavior and the nonlinear material behavior. For design the complex material behavior is normally simplified with a linear elastic model. However, these analysis are not accurate and very large partial safety factors need to be used. There has been done some research to obtain accurate material characteristics. In the M.Sc. project of P.H. van Asselt, bi-axial loading tests on PTFE coated fiberglass fabric have been performed in the Stevin Lab at the Faculty of Civil Engineering, Delft University of Technology.

Based on the experimental result several attempts have been made to model the fabric's stress-strain behavior. It was not successful to fit curved surfaces through the experimental data (P.H. van Asselt, 2007). When the strain range is decreased to a certain extent, the nonlinear behavior can be modeled as linearly. But this approach is still non-usable for an industrial application. Therefore the phenomenological approach was left and a nonlinear textile model has been developed (P.C.J. Hoogenboom, 2007). Based on the fibers' geometry and interaction mechanism, several equations are used to represent the nonlinear behavior of the fabric.

This model has been implemented successfully by Mr. P.H. van Asselt to analyze a square piece of fabric, which results in a reasonable accuracy for biaxial stress situations. Nonetheless, the convergence issues caused a large calculation time. Attempts to analyse a tent structure failed due to convergence problems.

In this report the applicability of this nonlinear textile model is improved. It is implemented in finite element software ANSYS (User Material Subroutine USERMAT) to analyze complex tent structures under realistic loading cases. Initial prestress on the membrane structure is added to the textile model by temperature increment. Multiple load steps are defined by a combination of the USERMAT and ANSYS script (APDL). Effort is also spent on convergence matters to make the calculation time acceptable for engineering application.

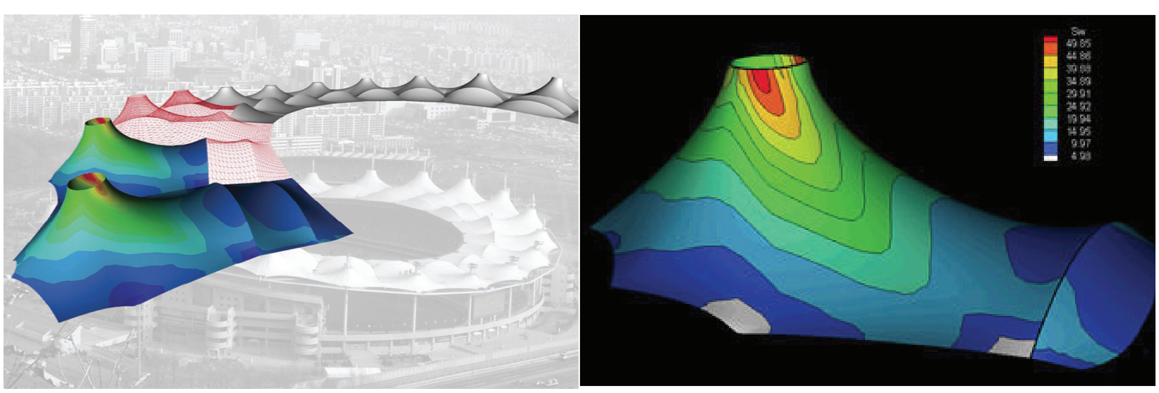


Figure 1 Munich Stadium, Frey Otto. FE model by Tensys Engineering/Analysis, www.tensys.com

1.2 PTFE coated fiberglass fabric

Fabric that is used for membrane structures is built up out of a woven structural base material. It is covered on both sides to protect it from water and pollutants. There are various ways to establish a coherent woven cloth. For structural use, basket bond and panama bond are normally used, as shown in Figure 2 (Houtman and Orpana, 2000).

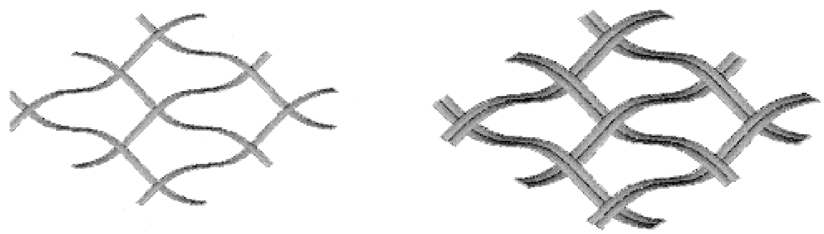


Figure 2 Basket bond and Panama bond (Houtman and Orpana, 2000)

Polytetrafluoroethene (PTFE) is a synthetic fluoropolymer which finds a lot of applications. The Teflon coated fiberglass fabric is the most permanent one of the architectural fabrics. After firstly employed for a roof in 1973, it has been widely used as coating material for lightweight structures.

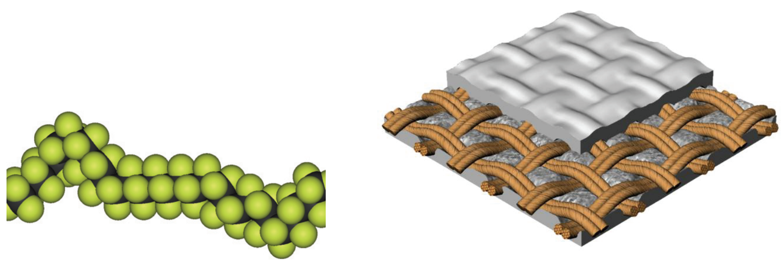


Figure 3 Polytetrafluoroethene molecule, www.wikipedia.org

The mechanical properties of PTFE coated fiberglass fabric is quite complex when compared with traditional building materials. The fabric is highly nonlinear, anisotropy and non-elastic. As can be seen from the test result in Figure 4, the fabric shows non-linear stress -strain behavior and various biaxial interactions for different stress ratios.

Figure 4 Strain measurements for load ratio 1:1 and 1:2 (P.H. van Asselt, 2007)

Because the fabric is built up out of woven base material, the material has two dominant head directions, warp and weft. The threads in the warp direction run straight in the weaving process, while the weft threads sneak around the warp threads, going over and underneath.

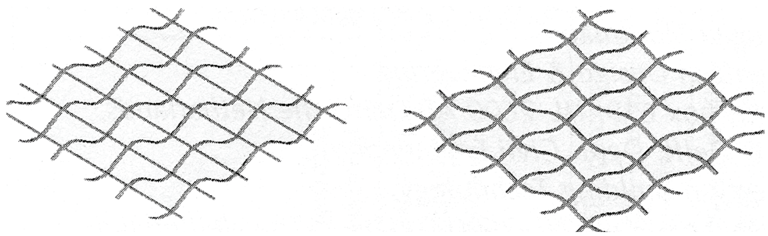


Figure 5 Warp and weft configuration before and after stressing (Houtman, 2000)

This configuration will cause a different strain in warp and weft direction. When the fabric is tensioned, there will be less deformation in warp direction than weft direction. Note that in this report this effect is not considered and it is assumed the properties for both warp and weft fibers are equivalent.

2 Structural Model for Textile

The nonlinear structural model for textile proposed by P.C.J. Hoogenboom, is based on the fiber interaction mechanism as shown in Figure 6.

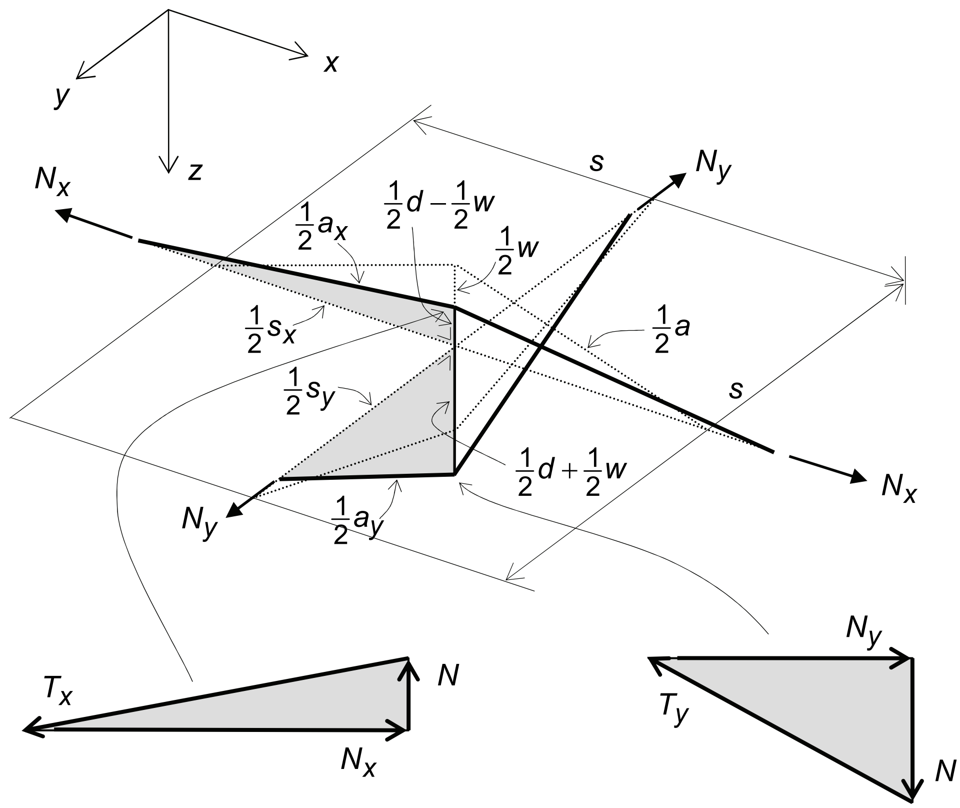


Figure 6 Theory of fiber interaction (Hoogenboom, 2007)

The kinematic equations can be expressed as:

$$\varepsilon_{xx} = \frac{s_x - s}{s}, \qquad \varepsilon_{yy} = \frac{s_y - s}{s} \tag{2.1}$$

$$a^2 = s^2 + d^2 \tag{2.2}$$

$$a_x^2 = s_x^2 + (d - w)^2 \tag{2.3}$$

$$a_y^2 = s_y^2 + (d + w)^2 \tag{2.4}$$

where

a	length of the undeformed wire
d	diameter of the undeformed wire
s	spacing of the undeformed wires
w	displacement of the crossing over point due to tension

a_x length of the stretched wire in the x-direction
a_y length of the stretched wire in the y-direction
s_x projection of a_x on the plane of the fiber
s_y projection of a_y on the plane of the fiber

The constitutive equation can be defined as:

$$T_x = EA\frac{a_x - a}{a} \qquad T_y = EA\frac{a_y - a}{a} \tag{2.5}$$

where
T_x , T_y tensile force in the wires
E Young's modulus
A area of a wire, $A = 1/4\pi d^2$

The equilibrium equations can be observed from Figure 6:

$$\frac{T_x}{N} = \frac{a_x}{d - w}, \qquad \frac{T_y}{N} = \frac{a_y}{d + w} \tag{2.6}$$

$$\frac{N_x}{T_x} = \frac{s_x}{a_x}, \qquad \frac{N_y}{T_y} = \frac{s_y}{a_y} \tag{2.7}$$

$$n_{xx} = \frac{N_x}{s}, \qquad n_{yy} = \frac{N_y}{s} \tag{2.8}$$

where
N vertical force at crossing over point
N_x , N_y external tensile force
n_{xx} , n_{yy} tensile stress

These nonlinear equations can be solved by a program, which uses iteration until the unbalance R in the force N is sufficiently small (see the MATLAB code on next page). This algorithm is used to calculate the fabric stress as a function of fabric strain. Three parameters, Young's modulus E, spacing s, and diameter d need to be properly set in order to describe the behavior of the PTFE coated fiberglass fabric tested at Stevin Lab. When the thickness of the fabric is 1.5 mm, theses parameters can be set as:

E = 77500 N/mm =51667 N/mm^2
s = 0.9 mm
d = 0.18 mm

MATLAB code I

```
% stress in membrane material
% written based on the MAPLE model created by Dr.ir. P.C.J. Hoogenboom, 2007
%
% -- input ------------------------------------------------------------------------------------------------
E   = 51667;                % [N/mm2]    Young's Modulus of the wires plus matrix
G   = 500;                  % [N/mm2]    shear modulus of the matrix
D   = 0.18;                 % [mm]       diameter of the wires
s   = 0.9;                  % [mm]       spacing of the wires (0<2d<s)
epsilonxx=0.003;            % [-]        strain in the x-direction
epsilonyy=0.003;            % [-]        strain in the y-direction
gammaxy=0.01;               % [-]        shear strain in the xy-direction
epsilonxxp=0.0;             % [-]        strain due to prestress in x-direction
epsilonyyp=0.0;             % [-]        strain due to prestress in y-direction
alpha=0.0;                  % [-]        linear temperature extension coefficient
dT=0.0;                     % [C]        temperature increase
% -- computation ------------------------------------------------------------------------------------------
%
a=sqrt(s^2+d^2);
A=1/4*pi*d^2;
sx=s*(1+epsilonxx+epsilonxxp);
sy=s*(1+epsilonyy+epsilonyyp);
w=0;
R=100;
while abs(R)>0.000001*a*d
   ax=sqrt(sx^2+(d-w)^2);
   ay=sqrt(sy^2+(d+w)^2);
   Tx=(ax/a-1-alpha*dT);
   Ty=(ay/a-1-alpha*dT);
   if Tx<0
     Tx=0;
   end
   if Ty<0
     Ty=0;
   end
   R=Tx*ay*(d-w)-Ty*ax*(d+w);
   w=w+R/a;
end
Tx=E*A*Tx;
Ty=E*A*Ty;
Nx=sx/ax*Tx;
Ny=sy/ay*Ty;
nxx=Nx/s;
nyy=Ny/s;
nxy=G*d*gammaxy;
%
% -- output ------------------------------------------------------------------------------------------------
nxx
nyy
nxy
```

3 Membrane Structural Analysis in ANSYS

To implement the nonlinear textile model in the analysis of tent structures is very chandelling, due to both nonlinear material behavior and geometrical nonlinear behavior of the membrane structure. The material model will be included in an external script which is programmed in the FORTRAN language (USERMAT), and then linked to the ANSYS software. In the calculation process, ANSYS will call the USERMAT to convert element strains into stresses for each iteration, and to obtain the tangent stiffness matrix (Jacobian matrix).

Since this process is quite complex, linking errors and convergence problems are likely to occur. In the following sections we will start with modeling and analyzing a structure with a linear isotropic material model, to verify that a proper model has been created, the right type of element has been chosen, and correct settings have been made. In small steps the modeling will be made more realistic.

3.1 Geometry

A cone-shape tent structure is modeled by ANSYS APDL commands. Two parameters *D* and *H* are defined to represent the span and the height. The surface of the tent can be easily generated by rotating a line about an axis.

The triangle type element is chosen to mesh the area. The size for meshing is 1000 mm and the entire model consists of 1536 elements.

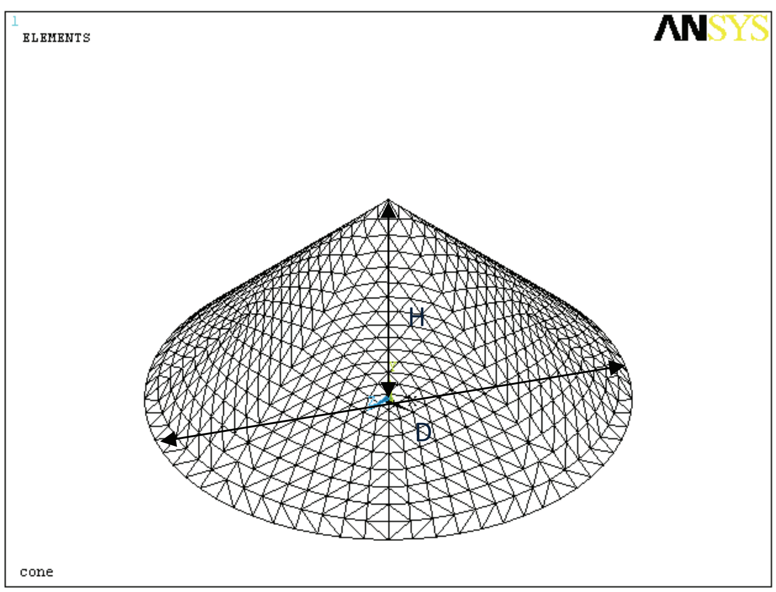


Figure 7 Modeling and Mesh

The model is fixed at the top node and all the nodes on the bottom surface. All the six displacement degrees of freedom and rotational degrees of freedom for each node are restrained.

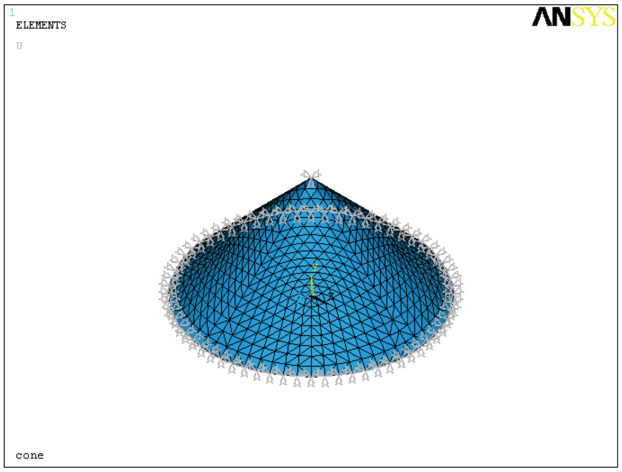


Figure 8 Boundary Conditions

APDL code

```
/UNITS, MPA
D=20000
H=10000

K, 1, 0, H, 0
K, 2, D/2, 0, 0
L, 1, 2
K, 1000, 0, 0, 0
AROTAT, 1, , , , , , 1, 1000, 360

MSHAPE, 1, 2D
MSHKEY, 0
AMESH, all

NSEL, S, LOC, Y, -0.1, 0.1
NSEL, A, NODE, , 0.5, 1.5
D, all, all
```

3.2 Element

ANSYS has a membrane element SHELL41 for membrane structure analysis. It is a 3-D element with membrane stiffness but no bending stiffness. There are three translational degrees of freedom at each node. It is suitable to model the membrane action of the PTFE coated fiberglass fabric when linear elastic material behavior is assumed.

However, in order to implement the textile model in ANSYS, the USERMAT must be created to provide the nonlinear constitutive law. This user subroutine is only applicable to elements from 18X family: LINK180, SHELL181, PLANE182/183, SOLID185/186/187 and BEAM188/189 (ANSYS Inc., 1999). Therefore SHELL181 will be used for the membrane analysis in this report. In case of an isotropic material model, the membrane element SHELL41 is also used to verify the membrane action of SHELL181 element.

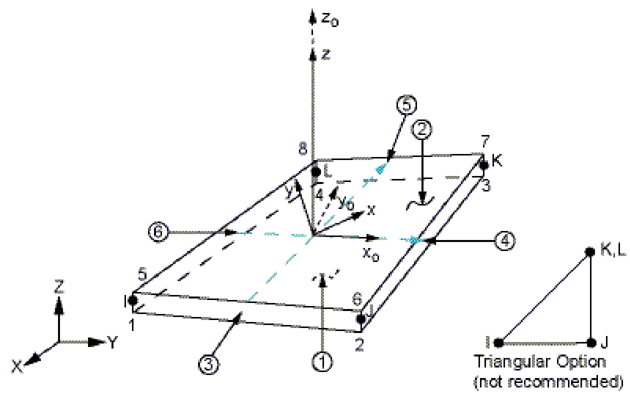


Figure 9 SHELL181 Geometry (ANSYS Inc., 2005)

SHELL181 is a 4-node element with six degrees of freedom at each node. The bending stiffness can be chosen to be activated or not. The geometry, node locations, and the coordinate system for this element are shown in Figure 9.

The thickness of the shell element is defined using Real Constants Sets. The thickness of the PTFE coated fiberglass fabric is set to be 1.5 mm.

```
T=1.5

ET, 1, SHELL181
KEYOPT, 1, 1
R, 1, T
```

3.3 Load and Prestress

An upward wind load perpendicular to the tent surface, with a magnitude of 2.0 kN/m^2 = 0.002 N/mm^2, is applied on the structure.

When a linear isotropic material model is used, the desired initial prestress can be produced by a temperature load in the tent model. In order to get an identical prestress state with the bi-axial test at Stevin Lab, the following prestress force will be used:

$$P = 500 \text{ N} / 18 \text{ cm} = 2.7778 \text{ N/mm} \quad (3.1)$$

Thhe thermal stress needed for this prestress is:

$$\sigma = P / \text{Thickness} = 2.7778 / 1.5 = 1.8519 \text{ MPa} \quad (3.2)$$

The thermal strain is:

$$\varepsilon = \sigma / E = \alpha\,(T_{reference} - T) \quad (3.3)$$

Therefore the temperature for prestress can be calculated by:

$$T = -\sigma / (E\ \alpha) + T_{reference} = -1.8519 / (1.0\text{e-}5 * E) = -185190/E \qquad (3.4)$$

where

α coefficient of thermal expansion, α = 1.0E-5

$T_{reference}$ reference temperature, $T_{reference} = 0$

3.4 Isotropic Material Model

As mentioned before, the analysis will start from using an isotropic material model. For small fabric strains, the linear simplification of a fabric behavior is applicable. Parameters for an isotropic model can be obtained by fitting a surface through the test data (up to 1% strain).

Based on the fitted surface, the Young's Modulus and Poisson's Ratio for this type of textile are obtained (P.H. van Asselt, 2007):

$$v = 0.90 \qquad (3.5)$$

$$E = 434561\ \text{N/m} / 1.5\text{mm} = 289.7073\ \text{MPa} \qquad (3.6)$$

Note that the maximum value of Poisson's Ratio in ANSYS is 0.5 thus $v = 0.49$ is used. This will lead to a different constitutive relation between stress and strain compared with the experimental data. However, this isotropic model is only made to compare with the nonlinear textile model (USERMAT) in next chapter and to verify the ANSYS nonlinear analysis.

Based on Equation (3.4) the initial temperature for prestress can be calculated as:

$$T = -\sigma / (E\ \alpha) + T_{reference} = -1.8579 / (1.0\text{e-}5 * 289.7073) = -639.23\ °\text{C} \qquad (3.7)$$

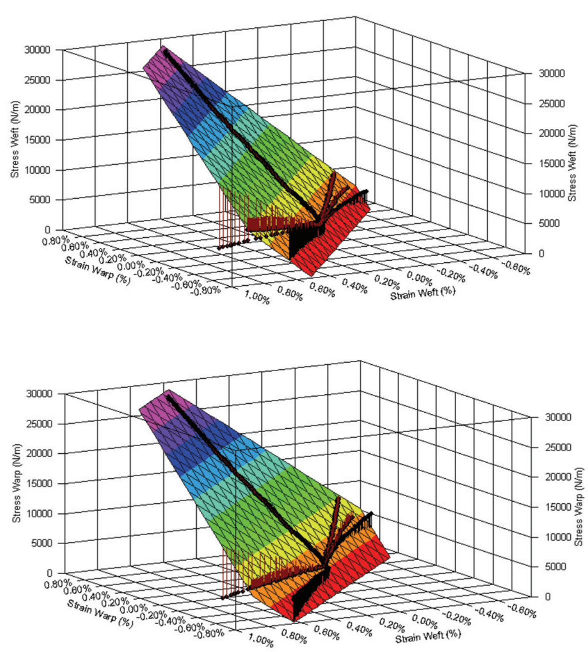


Figure 10 Fitted solution for Weft and Warp Direction (P.H. van Asselt, 2007)

3.5 Isotropic Material Model-Linear Analysis

A linear analysis is first performed using the isotropic model, see Figure 11 and 12. A large strain as much as 4.61% can be observed from the strain contours under wind loads. Obviously a geometrical nonlinear analysis is desirable for more accurate results.

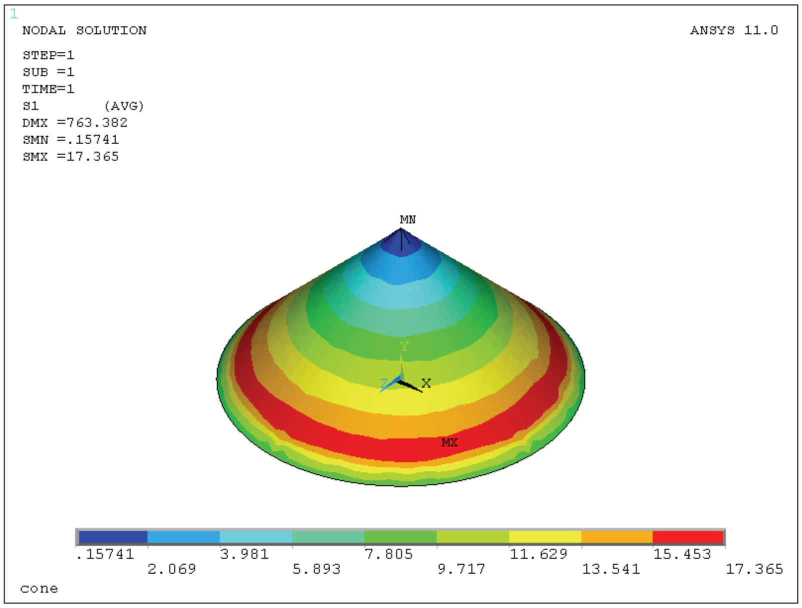


Figure 11 First Principal Stress Distribution, [MPa]

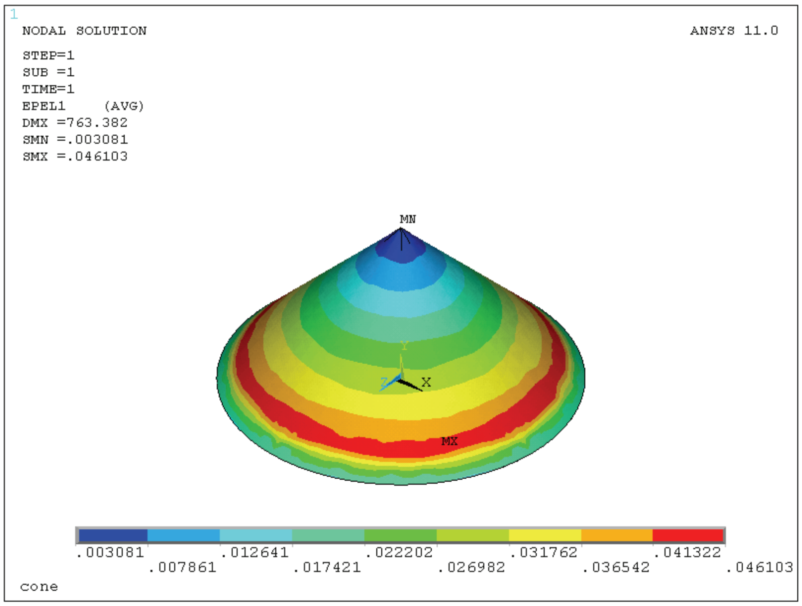


Figure 12 First Principal Strain Distribution

3.6 Isotropic Material Model-Geometric Nonlinear Analysis

ANSYS employs the ‘Newton-Raphson’ approach to solve nonlinear problems. In this approach, the load is first subdivided into a series of load increments and then each load step is subdivided into a series of substeps.

During the solution process, the Newton-Raphson method evaluates the out-of-balance load vector, which is the difference between the restoring forces (the loads corresponding to the element stresses) and the applied loads. The program then performs a linear solution, using the out-of-balance loads, and adds the resulting displacements to the current displacements. If convergence criteria are not satisfied, the out-of-balance load vector is reevaluated, the stiffness matrix is updated, and a new solution is obtained. This iterative procedure continues until the problem converges (ANSYS Inc., 2005).

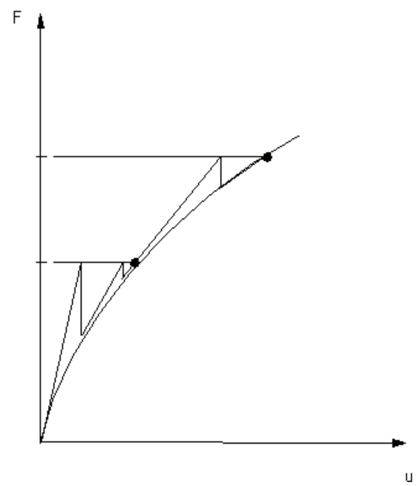


Figure 13 Newton-Raphson Approach

3.6.1 Settings for nonlinear analysis

ANSYS provides a number of commands to help the user get a proper result from a nonlinear analysis. Some convergence-enhancement and recovery features, such as automatic time stepping and bisection, can also be activated to help in case of converging problems. The commands and settings which have been used in the model are explained as below:

NLGEOM, on

When NLGEOM is active, the analysis will include large-deflection effects and the direction of a pressure load will change following the deflected element surface.

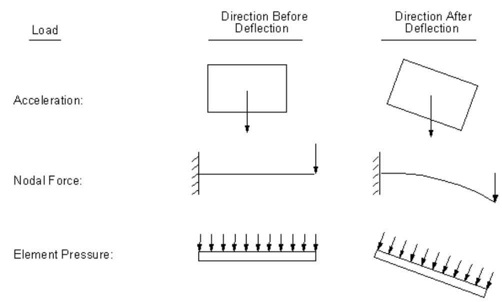


Figure 14 Load Direction before and after Deflection (ANSYS Inc., 2005)

```
SOLCONTROL, on
```

It will activate optimized defaults for a set of commands which are applicable to nonlinear solutions and some enhanced internal solution algorithms, including the stress stiffness effects, using a predictor on substeps after the first step, etc.

```
TIME, 1
NSUBST, 100, 10000, 1, on
AUTOTS, on
```

The time at the end of the load step is set to be 1. In case of other two commands are active at the same time, number of substeps to be used for this load step will firstly be set as 100. This will give the size of the first substep. Then automatic time stepping will take over the rest substeps using both time step prediction and time step bisection. The allowed number of substeps is between 1 and 10000.

```
NEQIT
```

The default maximum number of equilibrium iterations allowed in each substep is set between 15 and 26, when SOLCONTROL is set ON.

```
KBC, 0
```

KBC will specify stepped or ramped loading within a load step. If a load is ramped (0), then its value increases gradually at each substep, with the full value occurring at the end of the load step. If a load is stepped (1), then its full value is applied at the first substep and stays constant for the rest of the load step.

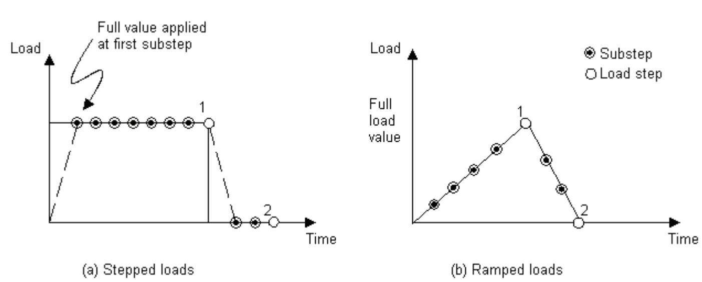


Figure 15 Stepped versus Ramped Loads (ANSYS Inc., 2005)

```
LNSRCH, auto
```

ANSYS will automatically switch line searching on and off between substeps of a load step as needed.

```
CNVTOL
```

This command will set convergence values for nonlinear analyses. When SOLCONTROL, on, the tolerance values default to 0.5% for force and moment, and 5% for displacement.

```
OUTRES, ALL, 1
OUTPR, ALL, 1
NCNV, 1
/GST, on
```

These commands specify the settings for solution tracking and solution data control.

3.6.2 Prestress by temperature load

To verify the initial stress caused by the temperature decrease, a nonlinear analysis with Temperature Load only is performed. The stress distribution is shown in Figure 16.

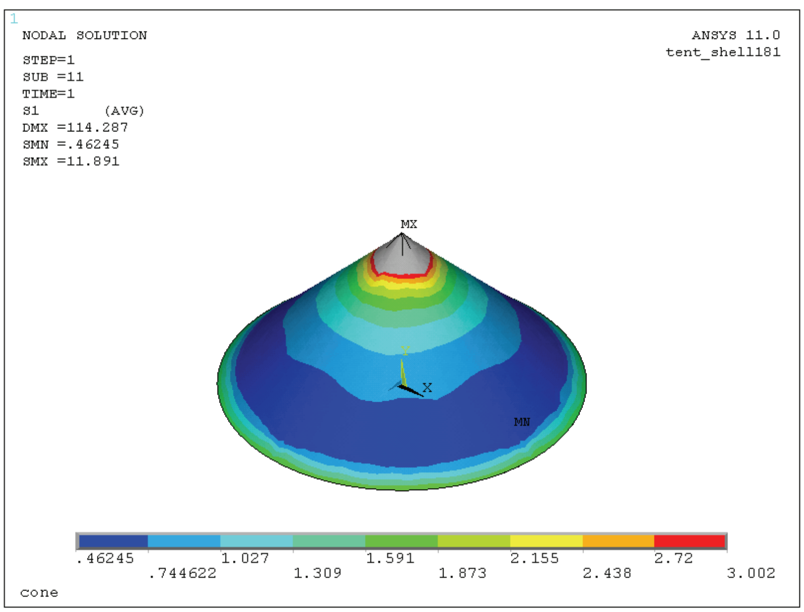


Figure 16 First Principal Stress Distribution, Temperature Load, [MPa]

Note that the stress caused by temperature is not uniformly distributed. For most parts of the tent surface, values between 0.4625 MPa and 1.591 MPa can be seen, while in the top of the cone it raises up to 11.891 MPa.

This prestress state is different than what is expected, which is 1.8519 MPa with a uniform distribution. A possible way to obtain the desired stress is to use a proper amplification factor, to have most parts of the tent surface under the desired stress state.

Figure 17 shows the initial stress distribution when amplification factor α_0 = 2.0 has been used.

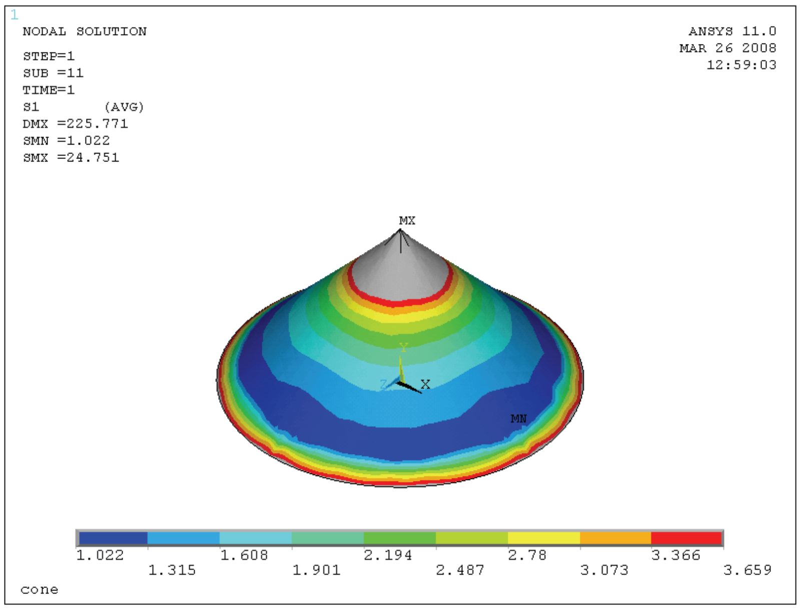


Figure 17 First Principal Stress Distribution, Temperature Load, Amplification Factor α=2.0, [MPa]

Alternatively, the element SHELL181 has an option to define initial stress by using user subroutine USTRESS. For this end the element key option, KEYOPT (10), should be set as 1 instead of the default value. The prestress will be read from the user subroutine USTRESS which is written in FOTRAN language. The effect of this method on tent structures needs a further study.

3.6.3 Nonlinear analysis under wind load

The geometric nonlinear analysis under wind load is performed with all the optimized settings. Force and moment convergence criteria are used with the default tolerance value 0.5%. ANSYS computes the convergence norm with corresponding convergence criterion for each iteration equilibrium. The solution is graphically tracked as shown in Figure 18. 11 substeps have been automatically chosen by ANSYS as shown in Table 1.

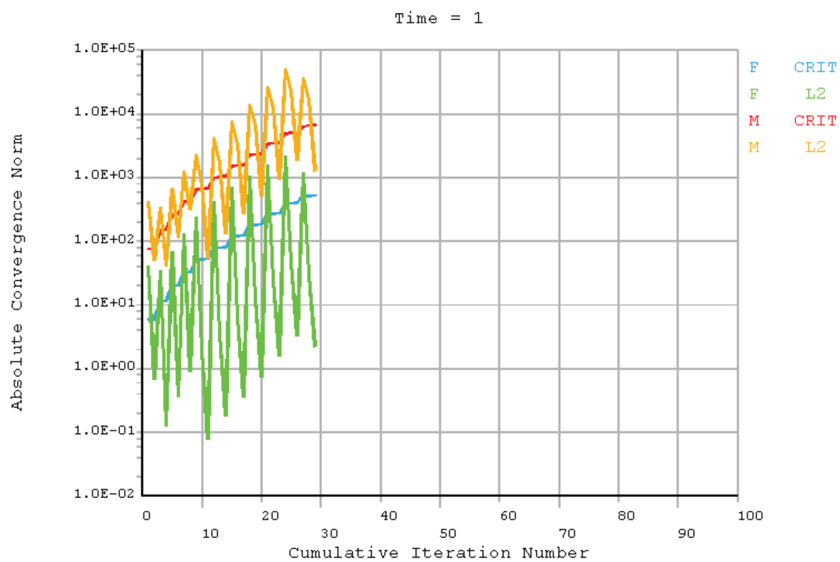


Figure 18 Convergence Norms Displayed by the Graphical Solution Tracking

Table 1 Substep and Size

Substep	Time	6	0.14188
1	1.00E-02	7	0.21781
2	2.00E-02	8	0.33172
3	3.50E-02	9	0.50258
4	5.75E-02	10	0.75887
5	9.13E-02	11	1

Element 50 is chosen for review and checking of the iteration history. The location of Element 50 is shown in Figure 19.

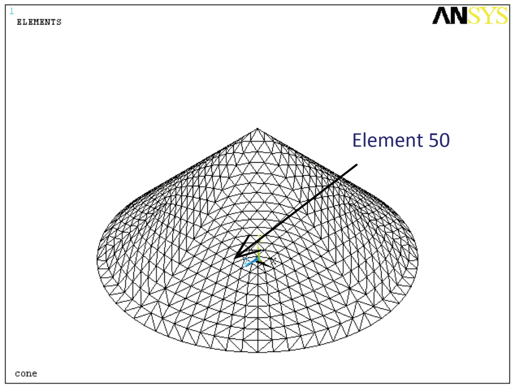


Figure 19 Location of Element 50

The first and second principal stress-strain relations at Element 50 are plotted in Figure 20 and 21. The stress-strain curve in Figure 21 is mainly caused by the large Poisson's ratio used in this case. The results show a smooth response history and thus a reasonable time step has been chosen.

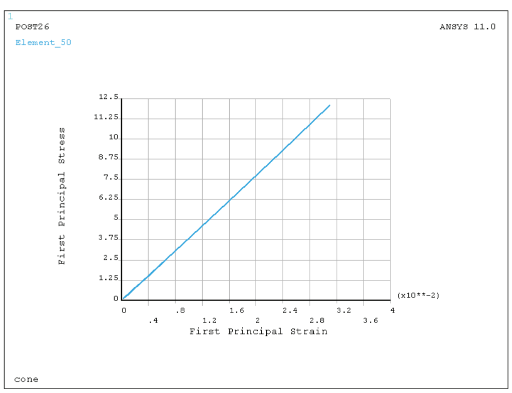


Figure 20 First Principal Stress-Strain at Ele.50

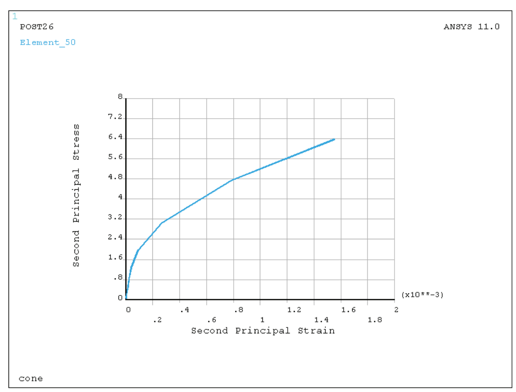


Figure 21 Second Principal Stress-Strain at Ele.50

The first principal stress and strain distributions are shown in Figure 22 and 23. The tent surface deform smoothly with a maximum strain of 3.14%. It decreases compared with the previous result from a linear analysis with a value of 4.61%. The maximum principal stress also decreases from 17.365 MPa to 12.485 MPa.

It is noted that the isotropic model is a linear simplification of fabric behavior up to 1% fabric strain, and the maximum strain obtained in Figure 23 is as much as 3.14%. So this material model is actually not applicable for such a large deflection and a nonlinear material model is desirable.

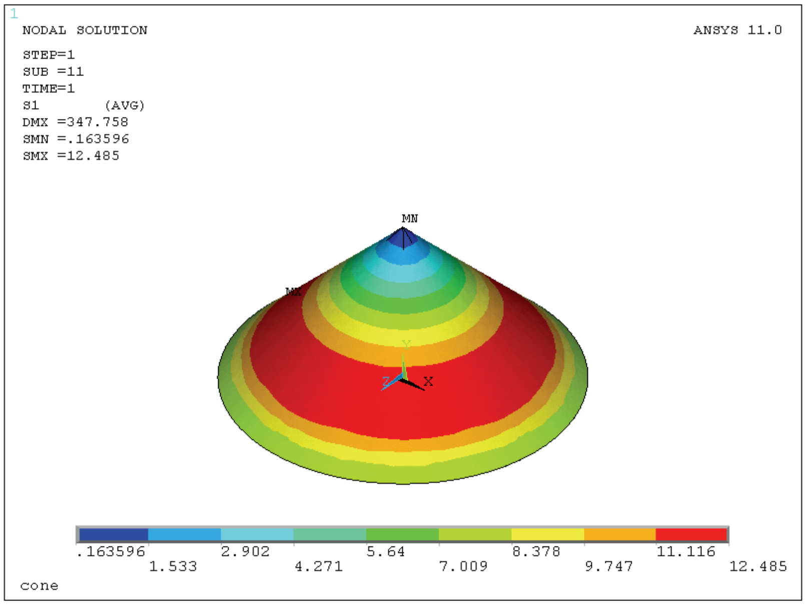


Figure 22 First Principal Stress Distribution under Wind Load, [MPa]

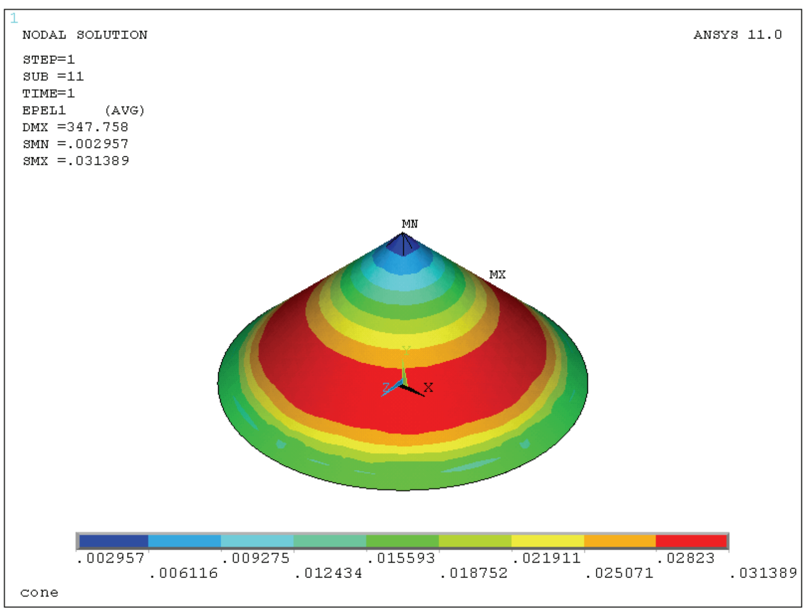


Figure 23 First Principal Strain Distribution under Wind Load

3.6.4 Verification by membrane element-SHELL41

The results from SHELL181 element can be verified by using a different ANSYS membrane element SHELL41. A similar nonlinear analysis is performed with this membrane element. The POST26 results of principal stress and strain components are printed for element 50. As can be seen in the tables below, the membrane element and SHELL181 element give closed results.

Table 2 Element 50 POST26 Output-SHELL41

ANSYS POST26 VARIABLE LISTING
SHELL41

TIME	1st principal stress	2nd principal stress	1st principal strain	2nd principal strain
1.00E-02	0.114008	5.60E-02	2.99E-04	3.56E-07
2.00E-02	0.228317	0.112649	5.98E-04	2.67E-06
3.50E-02	0.400454	0.198329	1.05E-03	7.27E-06
5.75E-02	0.659853	0.327719	1.72E-03	1.52E-05
9.13E-02	1.05126	0.523238	2.74E-03	2.80E-05
0.14188	1.64283	0.819294	4.28E-03	4.94E-05
0.21781	2.53691	1.26909	6.61E-03	8.98E-05
0.33172	3.88193	1.95522	1.01E-02	1.83E-04
0.50258	5.88141	3.00519	1.52E-02	4.26E-04
0.75887	8.79727	4.61258	2.26E-02	1.04E-03
1	11.4328	6.1469	2.91E-02	1.88E-03

Table 3 Element 50 POST26 Output-SHELL181

ANSYS POST26 VARIABLE LISTING
SHELL181

TIME	1st principal stress	2nd principal stress	1st principal strain	2nd principal strain
1.00E-02	0.114504	5.63E-02	3.00E-04	8.32E-07
2.00E-02	0.229276	0.112976	6.00E-04	2.17E-06
3.50E-02	0.402031	0.198314	1.05E-03	4.55E-06
5.75E-02	0.662746	0.327195	1.73E-03	8.46E-06
9.13E-02	1.05776	0.522439	2.77E-03	1.43E-05
0.14188	1.65881	0.819452	4.34E-03	2.29E-05
0.21781	2.57577	1.27373	6.74E-03	4.00E-05
0.33172	3.97191	1.97294	1.04E-02	9.22E-05
0.50258	6.07922	3.05581	1.58E-02	2.66E-04
0.75887	9.21176	4.74088	2.38E-02	7.84E-04
1	12.0965	6.37708	3.10E-02	1.55E-03

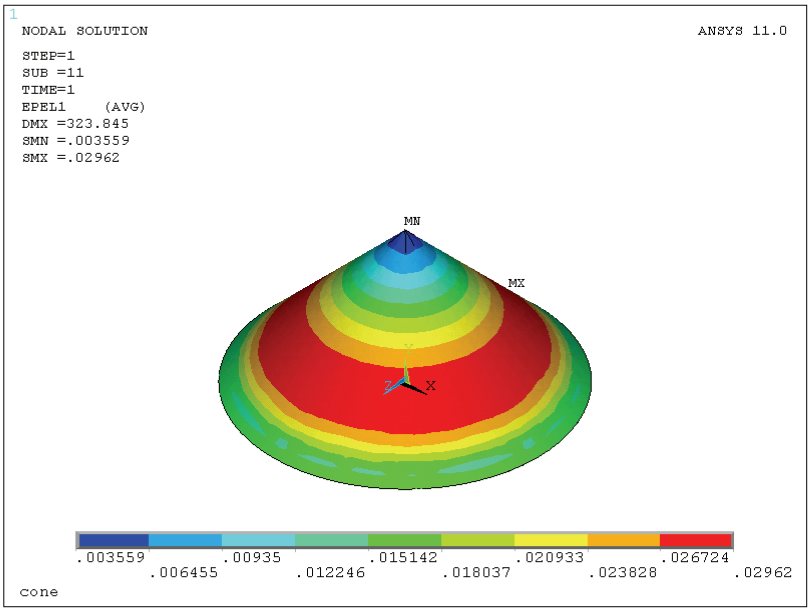


Figure 24 First Principal Stress Distribution under Wind Load, SHELL41, [MPa]

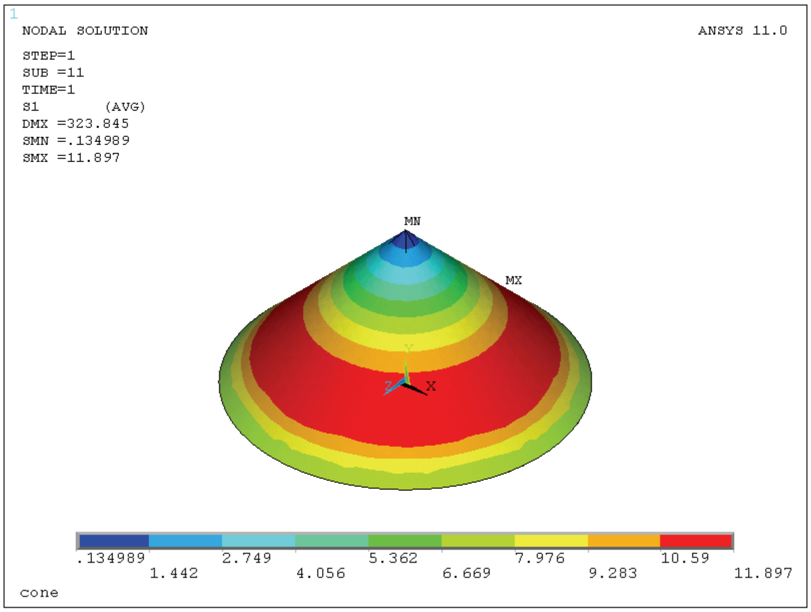


Figure 25 First Strain Distribution under Wind Load, SHELL41

4 ANSYS User Material Subroutine USERMAT

The nonlinear fabric model in Chapter 2 is programmed into an outer subroutine in the FORTRAN language. This user material routine USERMAT is an ANSYS user-programmable feature for use with 18x family elements. It allows users to write their own material constitutive equations.

4.1 Setting up a USERMAT

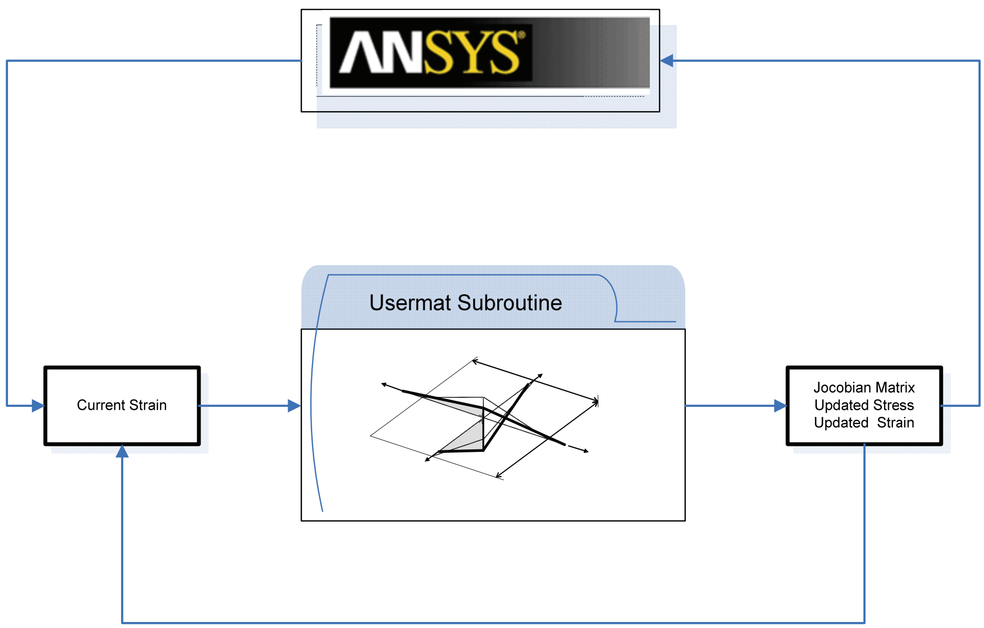


Figure 26 Iterative Process

The material model USERMAT defines the mechanical constitutive behavior based on the fiber interaction mechanism. For every Newton-Raphson iteration, USERMAT is called at every material integration point. ANSYS passes in stresses, strains, strain increments, and state variables at the beginning of the time increment. USERMAT then updates the strains and stresses as the output of the model. Material Jacobian matrix is also provided for convergence matters.

For SHELL181 element, a plane stress algorithm for the material constitutive integration must be used. In that case the total number of the stress or strain components at material point will be three. This will define the sizes for strain vector, strain increment vector, stress vector and Jacobian Matrix.

$$ncomp = 3 \quad (4.1)$$

At the beginning of each step, the strain will be updated as:

$$Strain\ (i) = Strain\ (i) + dSrain\ (i) \tag{4.2}$$

where

Strain Double precision array contains the total strains

dStrain Double precision array contains current strain increments

i i = 1, 2, … , ncomp

The updated strain vector *Strain (i)* will be used to calculate the stresses by using the fiber model in Chapter 2. The results are stored in the stress vector *Stress (i)*.

$$Stress\ (i) = f_i\ [\ Strain\ (1), Strain\ (2), \ldots, Strain\ (ncomp)\ ] \tag{4.3}$$

where

Stress Double precision array contains the stresses

$f_i\ [x]$ Nonlinear constitutive law for PTFE coated fiberglass fabric

In USERMAT the Jacobian matrix is also calculated and it is stored in *dsdeP1 (ncomp, ncomp)*. *dsdeP1 (i, j)* denote the change in the i-th stress component at the end of the time increment caused by a change of the j-th strain component. It will be used for convergence matters only, therefore it does not need to be the exact matrix but can be a general one.

Because ANSYS assumes that the element stiffness matrix is symmetric, therefore a symmetric material Jacobian matrix must be provided by USERMAT even it is unsymmetric. There are various ways to create an estimation of the exact Jacobian matrix. Three different approaches (Jacobian-I, Jacobian-II, Jacobian-III) have been studied in this report.

Jacobian-I is the simplest matrix which uses the constitutive law of linear elastic material. The matrix can be expressed as:

$$Jacobian\text{-}I = dsdeP1(ncomp,ncomp) = \begin{bmatrix} \frac{E}{1-v^2} & \frac{vE}{1-v^2} & 0 \\ \frac{vE}{1-v^2} & \frac{E}{1-v^2} & 0 \\ 0 & 0 & G \end{bmatrix}$$

Jacobian-II is calculated based on a two-step strain increment on the nonlinear fiber model. The matrix will be made symmetric by having *dsdeP1* (1,2) and *dsdeP1* (2,1) equal to their average value. In addition, Shear modulus *G* is used and four elements with zero value are assumed as shown in Figure 27.

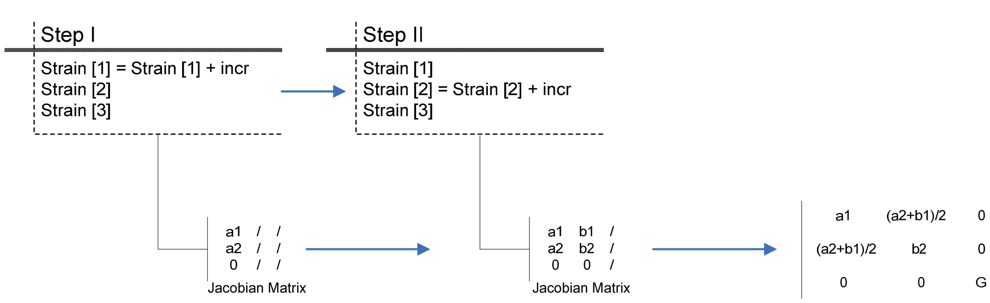


Figure 27 Jacobian Matrix-II

Jacobian-III is obtained by a three-step strain increment as shown in Figure 28, which might lead to a slower process but a more accurate matrix compared with other two.

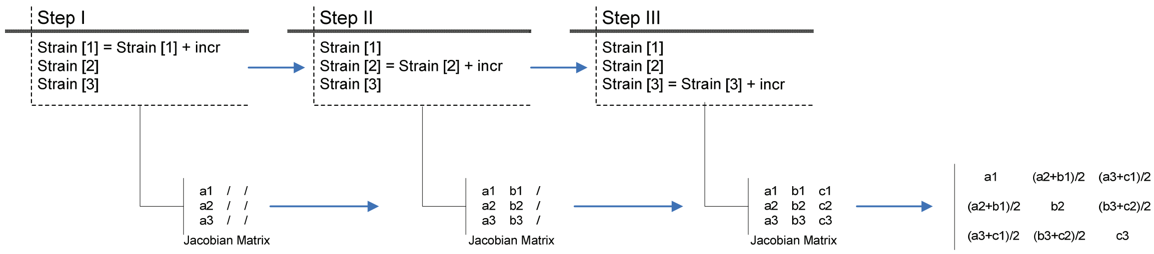


Figure 28 Jacobian Matrix-III

Based on these different approaches, several USERMATs have been created and tested. The Jacobian Matrix II shows to work well in most of the test cases, and the calculation time for a two-load-step case is no more than 10 minutes with a proper nonlinear setting. In case that Jacobian-I or Jacobian-III has been used, convergence problems were encountered for a few cases. However from this we cannot conclude that Jacobian-II is the most suitable solution for the nonlinear textile model. The nonlinear analysis is a complex process and many factors can influence the result.

In addition, for the PLANE element and SOLID element of the 18x family elements, unsymmetric Jacobian matrix is allowed when an element key option, KEYOPT (5) is set to be 1. This may be a possible approach if an unsymmetric Jacobian matrix is desirable in a future development.

4.2 Compiling and Linking USERMAT

After creating a USERMAT, the source files need to be compiled and linked to ANSYS. By using the Relink option from the ANS_ADMIN utility, ANSYS compiles all FORTRAN files and C files in a specified directory. Therefore the USERMAT must be placed in this folder. The procedure then loads all object files and the default ANSYS objects and libraries. At last a new executable file is created.

It took a lot time before we had the compiling and linking of USERMAT right and successful. A suitable version of Intel FORTRAN Compiler and C Compiler are required for this operation. The environment variables must be properly set. For convenience of a future work some tips on the USERMAT compiling and linking have been written and included in Appendix I.

The new executable file can be selected by running the ANSYS Product Launcher. USERMAT then is available to be used. To implement this material model, the TB USER command must be first issued in the ANSYS script (APDL) as:

```
TB, USER, matID, NTEMPS, NPTS
```

With this command the material reference number, number of temperature points and number of material constants at a given temperature point will be defined.

The material constants that will be used as input for USERMAT can be defined in APDL as

TBDATA, Starting Location, C1, C2, C3 …

The total number of material constants depends on the algorithm contained in USERMAT. In this report three different types of USERMAT has been included: a linear USERMAT, a nonlinear USERMAT without initial stress, and a nonlinear USERMAT with prestress and multiple load steps. They will require two (E, v), three (E, d, s) and five (E, d, s, a_1, a_2) material constants respectively.

4.3 Verification of USERMAT

A linear USERMAT is firstly written to verify the compiling and linking between USERMAT and ANSYS script. This USERMAT model consists of a linear material constitutive law and Jacobian-I matrix. The result is expected to be identical with the result in Chapter 3 where an ANSYS isotropic material model is used.

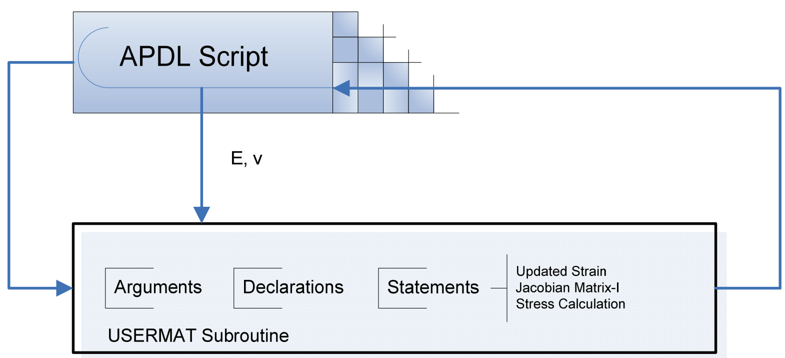


Figure 29 Iterative Process with a Linear USERMAT

The USERMAT statements cover the linear material model and the Jacobian Matrix-I, where strains and stresses will be updated and stored. The material constants are defined in APDL script in order to facilitate quick material adaptations without recompiling and relinking the USERMAT. Two material constants are needed for this model:

$$v = 0.49 \quad (4.4)$$

$$E = 289.7073 \text{ MPa} \quad (4.5)$$

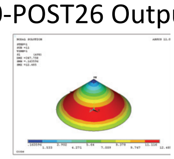
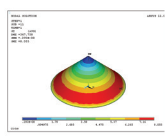
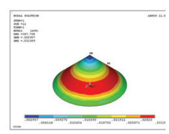
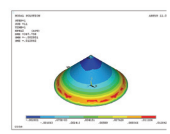
The POST26 results for Element 50 are listed in Table 4. It turns out to be exactly the same with previous results in Table 3. So the entire iterative process is well built.

The Statements in USERMAT, Linear constitutive law

```
get Young's modulus and Poisson's ratio
   young   = prop(1)
   posn    = prop(2)
   twoG    = young / (ONE+posn)

calculate elastic stiffness matrix (3d)
   c1 = ONE - posn * posn
   c2 = young / c1
   c3 = posn * c2
   dsdePl(1,1) = c2
   dsdePl(1,2) = c3
   dsdePl(1,3) = ZERO
   dsdePl(2,2) = c2
   dsdePl(2,3) = ZERO
   dsdePl(3,3) = HALF * twoG

   do i=1,ncomp-1
    do j=i+1,ncomp
     dsdePl(j,i)=dsdePl(i,j)
    end do
   end do

calculate elastic strain
   do i=1,ncomp
     wk1(i) = Strain(i) + dStrain(i)
   end do

update stresses
   stress(1) = wk1(1) * c2 + wk1(2) * c3
   stress(2) = wk1(1) * c3 + wk1(2) * c2
   stress(3) = wk1(3) * HALF * twoG
```

See Appendix I for the full USERMAT code.

Table 4 Post26 Output for Element 50 with a Linear USERMAT

Element 50-POST26 Output-Linear USERMAT				
				
TIME	1st stress	2nd stress	1st strain	2nd strain
1.00E-02	0.114504	5.63E-02	3.00E-04	8.32E-07
2.00E-02	0.229276	0.112976	6.00E-04	2.17E-06
3.50E-02	0.402031	0.198314	1.05E-03	4.55E-06
5.75E-02	0.662746	0.327195	1.73E-03	8.46E-06
9.13E-02	1.05776	0.522439	2.77E-03	1.43E-05
0.14188	1.65881	0.819452	4.34E-03	2.29E-05
0.21781	2.57577	1.27373	6.74E-03	4.00E-05
0.33172	3.97191	1.97294	1.04E-02	9.22E-05
0.50258	6.07922	3.05581	1.58E-02	2.66E-04
0.75887	9.21176	4.74088	2.38E-02	7.84E-04
1	12.0965	6.37708	3.10E-02	1.55E-03

5 Analysis with Nonlinear USERMAT Model

Building and using a nonlinear fiber USERMAT subroutine with ANSYS requires some caution in the analysis. To keep insight in the process and to have the behavior of the routine monitored, the USERMAT model is adapted and improved at small steps. During the study different versions of USERMAT, from the simple linear material model to the nonlinear fiber interaction model with prestress and multiple load steps, have been written, tested and implemented in the tent structural analysis. Two of them will be introduced in this chapter: USERMAT-I and USERMAT-II.

Both of this two models are based on fiber interaction mechanism as explained in Chapter2 (Figure 6) and the Jacobian-II matrix as explained in Chapter 4 is used. In USERMAT-I the prestress in tent structures is not evolved and it is linked to an APDL that only one load step is defined; while USERMAT-II includes temperature variables to introduce initial stresses and is combined with a multiple load-step APDL script.

In addition, different convergence features have be implemented to evaluate their effect. In case of USERMAT-I, automatic time stepping and bisection are activated, while fixed steps are used for USERMAT-II.

5.1 USERMAT-I

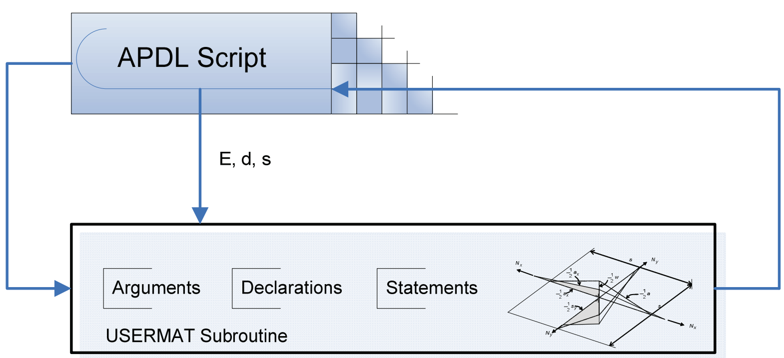


Figure 30 USERMAT Model-I

USERMAT-I consists of the nonlinear material model that uses several equations and iteration to represent the nonlinear fiber interaction mechanism as mentioned in Chapter 2. It requires three material constants, Young's modulus, diameter and spacing of the undeformed wires, which will be provided by APDL.

$$E = 77500 \text{ N/mm} = 51667 \text{ N/mm}^2 \quad (5.1)$$

$$d = 0.18 \text{ mm} \quad (5.2)$$

$$s = 0.9 \text{ mm} \quad (5.3)$$

The Statements used in USERMAT-I to update strains and calculate stresses are shown as follows:

The Statements in USERMAT-I, Nonlinear fiber interaction model

```
bisect/cut
    keycut   = 0

get material constants
    young = prop(1)
    d = prop(2)
    s = prop(3)
    G = 500.d0

get initial stress, diameter and spacing
    epsxxp=ZERO
    epsyyp=ZERO
    incr=ONEDM01

update strains
    do i=1,ncomp
      wk1(i) = Strain(i) + dStrain(i)
    end do

computation
    a=sqrt(s**TWO+d**TWO)
    Ad=FORTH*PI*(d**TWO)
      sx=s*(ONE+wk1(1)+epsxxp)
      sy=s*(ONE+wk1(2)+epsyyp)
      w=ZERO
      R=100.0d0
      do while (abs(R) .gt. ONEDM04*a*d)
        ax=sqrt(sx**TWO+(d-w)**TWO)
        ay=sqrt(sy**TWO+(d+w)**TWO)
        Tx=(ax/a-ONE)
        Ty=(ay/a-ONE)

      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif

        R=Tx*ay*(d-w)-Ty*ax*(d+w)
        w=w+R/a
      end do

      Tx=young*Ad*Tx
      Ty=young*Ad*Ty
      Nx=(sx*Tx/ax)
      Ny=(sy*Ty/ay)

      nxxp=Nx/s
      nyyp=Ny/s
      nxyp=G*wk1(3)*d

update stresses
    stress(1)=nxxp
    stress(2)=nyyp
    stress(3)=nxyp
```

The Jacobian-II matrix is used for convergence matters with a two-step strain increment as shown in Figure 27. A proper length for substep must be defined in APDL to assist in a fast convergence during the calculation.

The Code for calculation of Jacobian-II

```
add small strain increment in x direction
    epsxxinc=wk1(1)+incr

    a=sqrt(s**TWO+d**TWO)
    Ad=FORTH*PI*(d**TWO)
      sx=s*(ONE+epsxxinc+epsxxp)
      sy=s*(ONE+wk1(2)+epsyyp)
      w=ZERO
      R=100.0d0
      do while (abs(R) .gt. ONEDM04*a*d)
        ax=sqrt(sx**TWO+(d-w)**TWO)
        ay=sqrt(sy**TWO+(d+w)**TWO)
        Tx=(ax/a-ONE)
        Ty=(ay/a-ONE)

      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif

        R=Tx*ay*(d-w)-Ty*ax*(d+w)
        w=w+R/a
```

```
    end do

    Tx=young*Ad*Tx
    Ty=young*Ad*Ty
  Nx=(sx*Tx/ax)
    Ny=(sy*Ty/ay)

  nxx=Nx/s
  nyy=Ny/s

  dsdePl(1,1)=(nxx-nxxp)/incr
  dsdePl(2,1)=(nyy-nyyp)/incr
  dsdePl(3,1)=0

add small strain increment in y direction
  epsyyinc=wk1(2)+incr

  a=sqrt(s**TWO+d**TWO)
    Ad=FORTH*PI*(d**TWO)
    sx=s*(ONE+wk1(1)+epsxxp)
    sy=s*(ONE+epsyyinc+epsyyp)
    w=ZERO
    R=100.0d0
    do while (abs(R) .gt. ONEDM04*a*d)
      ax=sqrt(sx**TWO+(d-w)**TWO)
      ay=sqrt(sy**TWO+(d+w)**TWO)
      Tx=(ax/a-ONE)
    Ty=(ay/a-ONE)
  if (Tx .LT. 0) then
    Tx=0
  endif
  if (Ty .LT. 0) then
    Ty=0
  endif

    R=Tx*ay*(d-w)-Ty*ax*(d+w)
      w=w+R/a
  end do

    Tx=young*Ad*Tx
    Ty=young*Ad*Ty
    Nx=(sx*Tx/ax)
    Ny=(sy*Ty/ay)

  nxx=Nx/s
  nyy=Ny/s

  dsdePl(1,2)=(nxx-nxxp)/incr
  dsdePl(2,2)=(nyy-nyyp)/incr
  dsdePl(3,2)=0
  dsdePl(2,1)=dsdePl(1,2)
  dsdePl(1,3)=0
  dsdePl(2,3)=0
  dsdePl(3,3)=G*d
```

This material model is implemented in the analysis of the tent structure where an upward wind load is applied. The prestress is not included within this model.

$$P = 2 \text{ kN/m}^2 \tag{5.4}$$

The commands in APDL to provide material constants and to link with USERMAT-I are:

```
MAT, 1
TB, USER, 1, 1, 3
TBTEMP, 1.0
TBDATA, 1, 51667, 0.18, 0.9
TB, STATE, 1, , 6
```

In the process of the nonlinear analysis, full Newton-Raphson is used and the automatic time stepping, bisection and line search are activated:

```
NLGEOM, on
SOLCONTROL, on
TIME, 1
NSUBST, 100, 10000, 1, on
AUTOTS, on
KBC, 0
LNSRCH, auto
EQSLV, front
NROPT, full
```

5.2 USERMAT-I Result

USERMAT-I has been implemented in the analysis of the tent structure modeled in Chapter 3. During the process, 193 substeps have been defined by ANSYS, which leads to a large amount of calculation time (more than eight hours). This is probably because of the automatic time stepping and bisection used for the computation. In the second model USERMAT-II a fixed step will be used to evaluate the effect.

Element 50 is chosen again to check the result. See Figure 19 for the location of this element. The stress-strain relation is plotted as shown in Figure 31.

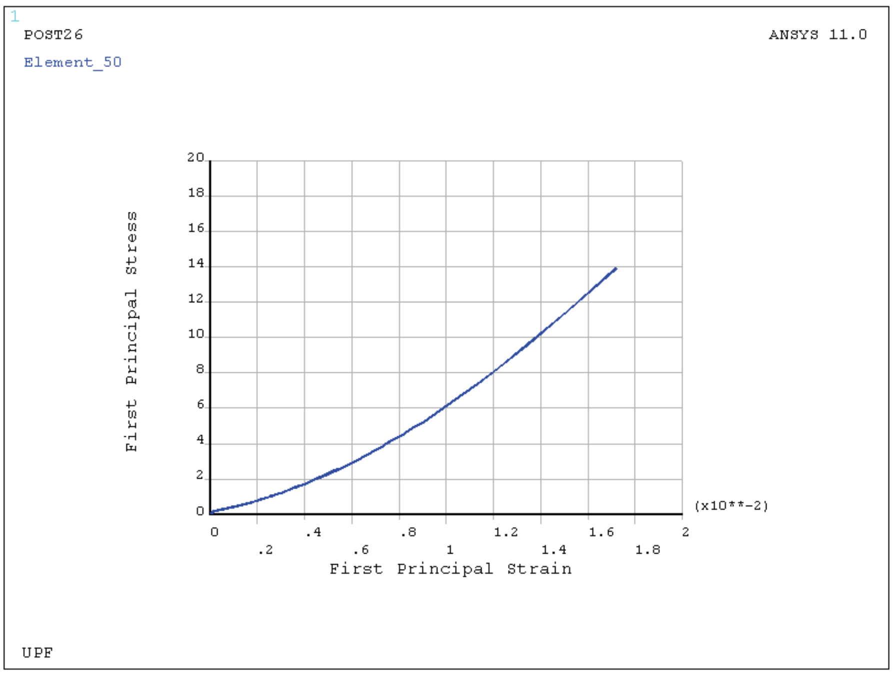


Figure 31 First Principal Stress-Strain for Element 50

The stresses ratio for element 50 after the final step is:

$$n_{ele50} = \frac{\sigma_1}{\sigma_2} = \frac{13.9362}{8.9954} = 1.55 \quad (5.5)$$

Figure 31 shows a good agreement with the curve obtained from the bi-axial loading test at Stevin Lab when a load ratio 2:1 is applied.

The stress and strain distributions are shown in Figure 32-35.

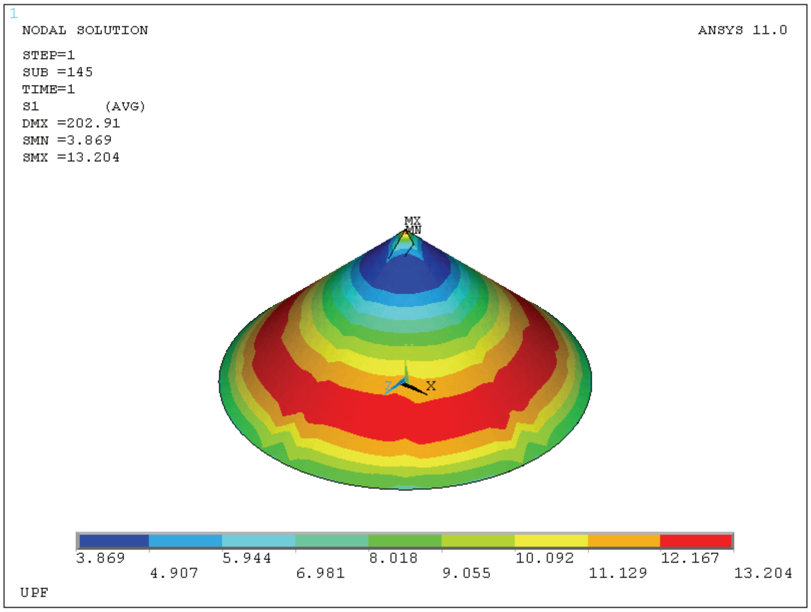


Figure 32 First Principal Stress Distribution, USERMAT-I, Wind Load, [MPa]

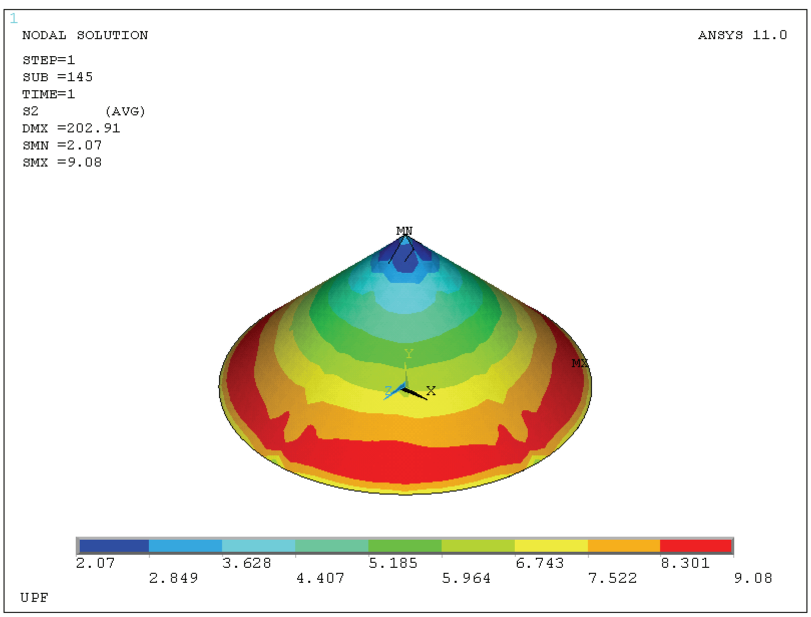


Figure 33 Second Principal Stress Distribution, USERMAT-I, Wind Load, [MPa]

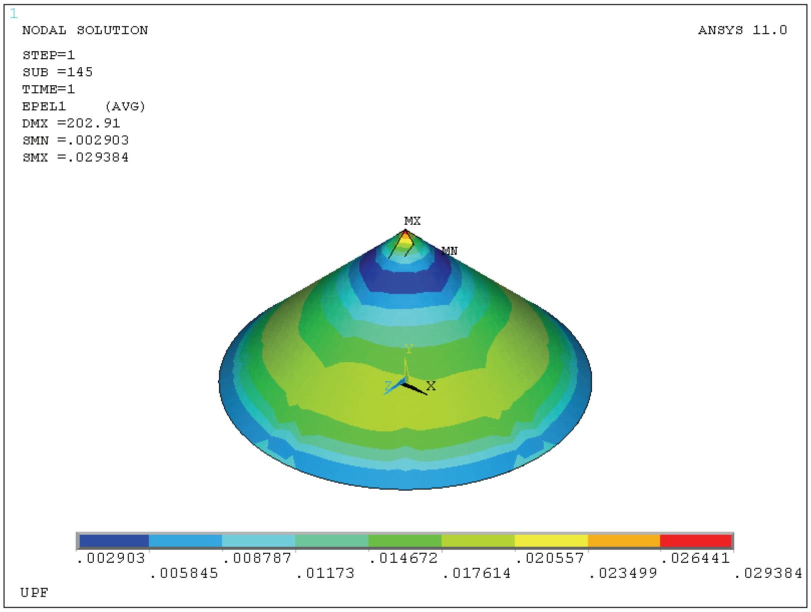


Figure 34 First Principal Strain Distribution, USERMAT-I, Wind Load

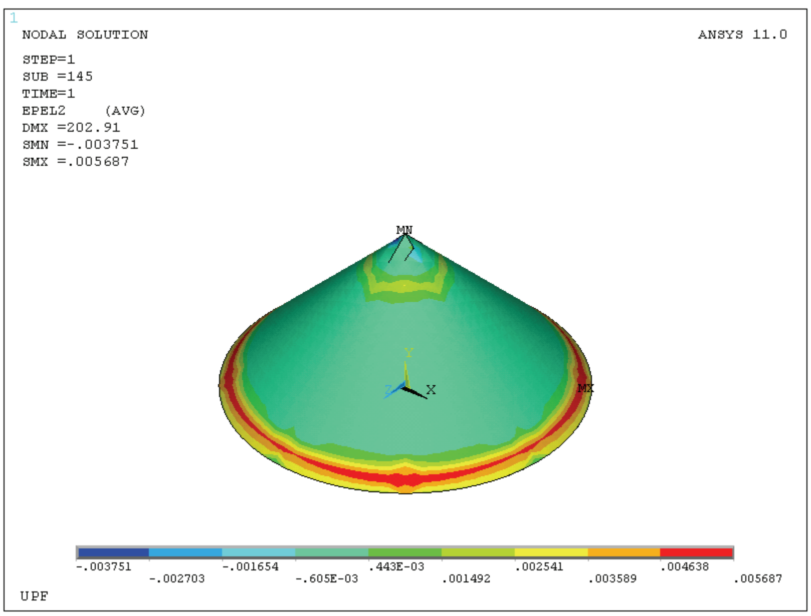


Figure 35 Second Principal Strain Distribution, USERMAT-I, Wind load

5.3 USERMAT-II

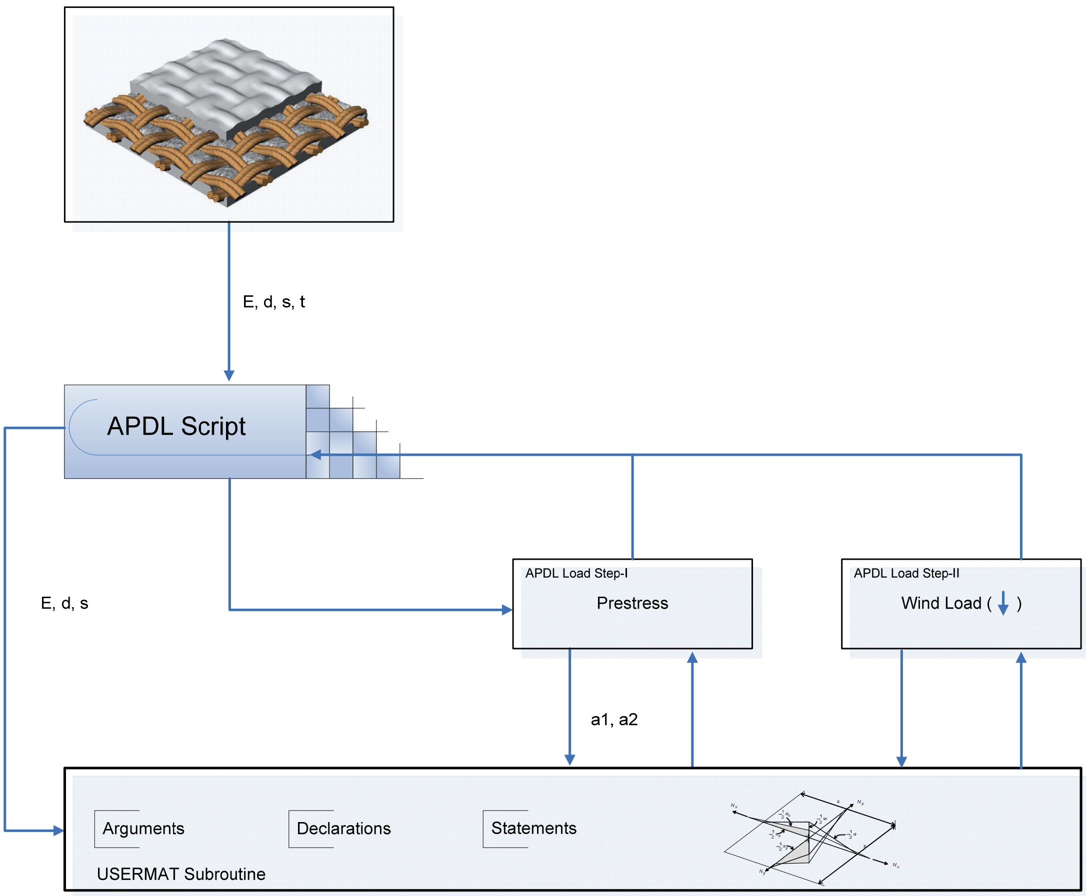


Figure 36 Analysis Process with USERMAT-II

USERMAT-II consists of the same material model as USERMAT-I, but it requires five material constant from APDL. Except for the Young's modulus, diameter and spacing of the undeformed wires, two prestress coefficients a_1 and a_2 are needed to apply the initial stress on the tent structure.

The initial strain caused by prestress within each iteration will be calculated by USERMAT-II as:

$$\begin{aligned} \varepsilon_{prestress,1} &= a_1 \cdot (temp + dtemp) \\ \varepsilon_{prestress,2} &= a_2 \cdot (temp + dtemp) \end{aligned} \tag{5.6}$$

where the coefficients a_1 and a_2 can be calculated as:

$$a_1 = \frac{\varepsilon_{prestress,1,total}}{\Delta T}$$
$$a_2 = \frac{\varepsilon_{prestress,2,total}}{\Delta T} \qquad (5.7)$$

ΔT is the temperature increment defined by APDL. $\varepsilon_{prestress,1,total}$ and $\varepsilon_{prestress,2,total}$ are the total initial strains caused by the prestress and can be calculated using the MATLAB Code II.

The Statements used in USERMAT-II to update the strains, stresses, as well as the temperature vector and prestress are shown as follows:

The Statements in USERMAT-II, Nonlinear model with prestress

```
bisect/cut
    keycut   = 0

get modulus
    young = prop(1)
    G = 500.d0

get diameter and spacing
    d=prop(2)
    s=prop(3)

initial strain by prestress
    c1=temp+dtemp
    epsxxp=prop(4)*c1
    epsyyp=prop(5)*c1
    incr=ONEDM01

update strains
    do i=1,ncomp
      wk1(i) = Strain(i) + dStrain(i)
    end do

computation
    a=sqrt(s**TWO+d**TWO)
    Ad=FORTH*PI*(d**TWO)
      sx=s*(ONE+wk1(1)+epsxxp)
      sy=s*(ONE+wk1(2)+epsyyp)
      w=ZERO
      R=100.0d0
      do while (abs(R) .gt. ONEDM04*a*d)
        ax=sqrt(sx**TWO+(d-w)**TWO)
        ay=sqrt(sy**TWO+(d+w)**TWO)
        Tx=(ax/a-ONE)
        Ty=(ay/a-ONE)

    if (Tx .LT. 0) then
      Tx=0
    endif
    if (Ty .LT. 0) then
      Ty=0
    endif

        R=Tx*ay*(d-w)-Ty*ax*(d+w)
        w=w+R/a
      end do

      Tx=young*Ad*Tx
      Ty=young*Ad*Ty
      Nx=(sx*Tx/ax)
      Ny=(sy*Ty/ay)

      nxxp=Nx/s
      nyyp=Ny/s
      nxyp=G*wk1(3)*d

update stresses
    stress(1)=nxxp
    stress(2)=nyyp
    stress(3)=nxyp
```

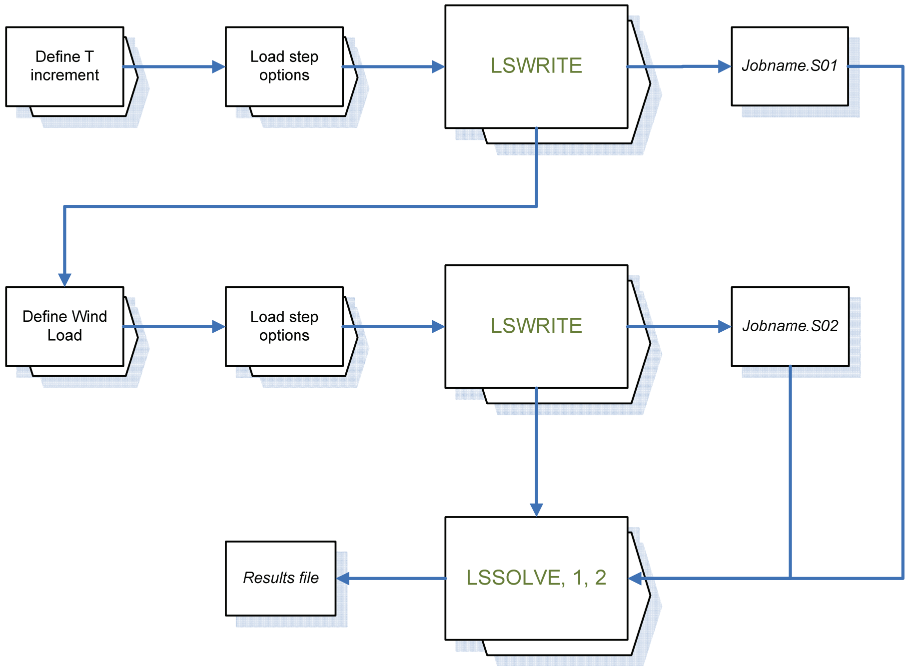


Figure 37 Multiple Load steps in APDL

An APDL script that can handle multiple load steps is constructed for USERMAT-II, as shown in Figure 37. The data for each load step is stored in the load step file Jobname.S01, Jobname.S02, etc. and will be read sequentially for solution. During the computation process, USERMAT-II is called at every material integration point. Results for each substep are written into output files for post processing. For a complete code see Appendix IV.

<u>MATLAB code II</u>

```
% strain in membrane material
% written based on the MAPLE model created by Dr.ir. P.C.J. Hoogenboom, 2007
% -- input ------------------------------------------------------------------------
E   = 51667;          % [N/mm2]   Young's Modulus of the wires plus matrix
d   = 0.18;           % [mm]      diameter of the wires
s   = 0.9;            % [mm]      spacing of the wires (0<2d<s)
nxx = 2.7778;         % [N/mm]    initial force in the x-direction
nyy = 2.7778;         % [N/mm]    initial force in the y-direction
% -- computation ------------------------------------------------------------------
a=sqrt(s^2+d^2);
A=1/4*pi*d^2;
Nx=nxx*s;
Ny=nyy*s;
if Nx<0
   Nx=0;
end
if Ny<0
   Ny=0;
end
w=d*(Nx-Ny)/(Nx+Ny+1e-8);
Tx=Nx;
Ty=Ny;
for i=1:1:4
   ax=a*(1+Tx/(E*A));
   ay=a*(1+Ty/(E*A));
   sx=sqrt(ax^2-(d-w)^2);
   sy=sqrt(ay^2-(d+w)^2);
   w=d*(Nx*sy-Ny*sx)/(Nx*sy+Ny*sx+1e-8);
   Tx=ax/sx*Nx;
   Ty=ay/sy*Ny;
end
epsilonxxp=sx/s-1;
epsilonyyp=sy/s-1;
% -- output ---------------------------------------
epsilonxxp
epsilonyyp
```

5.4 USERMAT-II Load Step I Prestress

From Equation (3.2), the desired initial prestress is 1.8519 MPa. Then the initial strains can be calculated by using MATLAB Code II in Section 5.3. The results are:

$$\varepsilon_{prestress,1,total} = 0.2\%$$
$$\varepsilon_{prestress,2,total} = 0.2\% \quad (5.8)$$

The temperature increment ΔT is defined as 10. From Equation (5.7) the prestress coefficients a_1 and a_2 can be calculated as:

$$a_1 = \frac{\varepsilon_{prestress,1,total}}{\Delta T} = 0.0002$$
$$a_2 = \frac{\varepsilon_{prestress,2,total}}{\Delta T} = 0.0002 \quad (5.9)$$

Having obtained all the parameters, the APDL script which provides material constants can be defined as:

```
MAT, 1
TB, USER, 1, 1, 5
TBTEMP, 1.0
TBDATA, 1, 51667, 0.18, 0.9, 0.0002, 0.0002
TB, STATE, 1, , 6
```

The stress distribution after first load step (prestress) is shown in Figure 38.

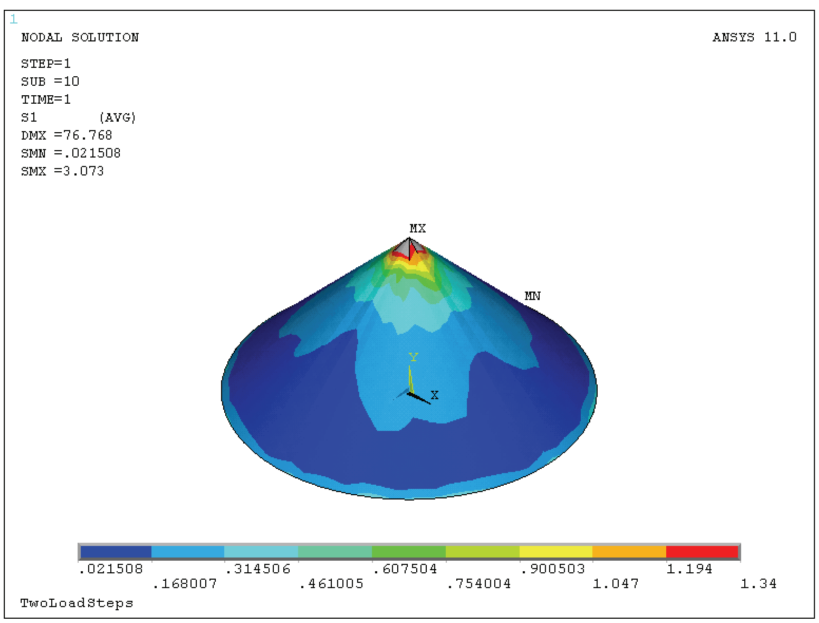


Figure 38 First Principal Stress Distribution, USERMAT-II, Initial Stress, [MPa]

A non-uniform stress distribution can be seen in the contour and most parts of the tent surface are under a stress level that is smaller than the expected one, as what has been observed in a linear model shown in Figure 16. Here an amplification factor α_0 is also used to obtain the desired prestress level.

Figure 39 shows the stress distribution when the amplification factor α_0 = 5.0 has been used.

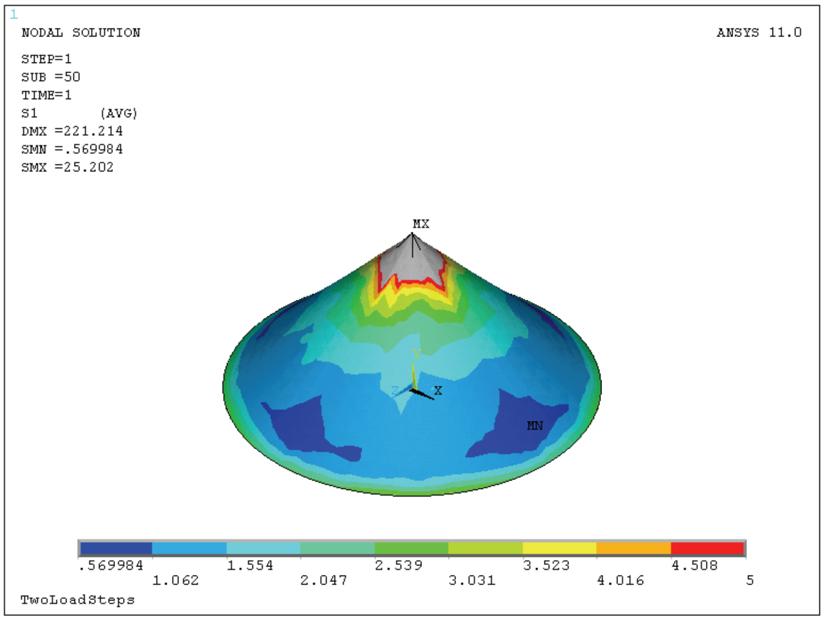


Figure 39 First Principal Stress Distribution, Initial Stress, Amplification Factor α =5.0, [MPa]

In case of α_0 = 5.0, the prestress coefficients a_1 and a_2 are:

$$a_1 = \alpha_0 \frac{\varepsilon_{prestress,1,total}}{\Delta T} = 0.001$$
$$a_2 = \alpha_0 \frac{\varepsilon_{prestress,2,total}}{\Delta T} = 0.001 \qquad (5.10)$$

The APDL that provides material constants should be redefined as:

```
MAT, 1
TB, USER, 1, 1, 5
TBTEMP, 1.0
TBDATA, 1, 51667, 0.18, 0.9, 0.001, 0.001
TB, STATE, 1, , 6
```

5.5 USERMAT-II Load Step II Wind Load

A downward wind load is applied on the tent structure as the second-step load.

$$P = \text{-1 kN/m}^2 \tag{5.11}$$

Since the automatic time stepping and bisection used in USERMAT-I have caused a large amount of calculation time, fixed steps are used in USERMAT-II. Different lengths of substep have been tested from 10 to 60. It is found that the most efficient way to obtain the nonlinear result is to define 10 fixed substeps for each load step when the meshing type is specified as Free Mesh. It takes approx. five minutes for each load step and the entire computational process can be finished within 10 minutes.

While a Map Mesh is specified, it causes some convergence problem in case of 10 fixed substeps. A number around 50 is suggested for the tent model tested in this report. The calculation time increased consequently to four hours. One advantage using a Map Mesh is that the result is much smoother compared with the result from a Free Mesh.

The stress and strain distributions after the final load steps are shown in Figure 40-43. The displacements are magnified by a factor in order to show the effect.

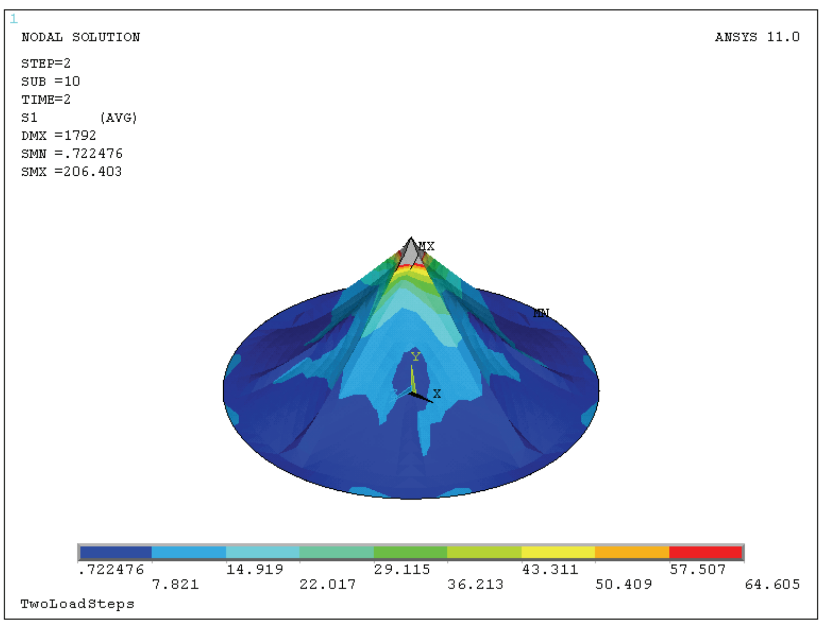


Figure 40 First Principal Stress Distribution, USERMAT-II, Final Step, [MPa]

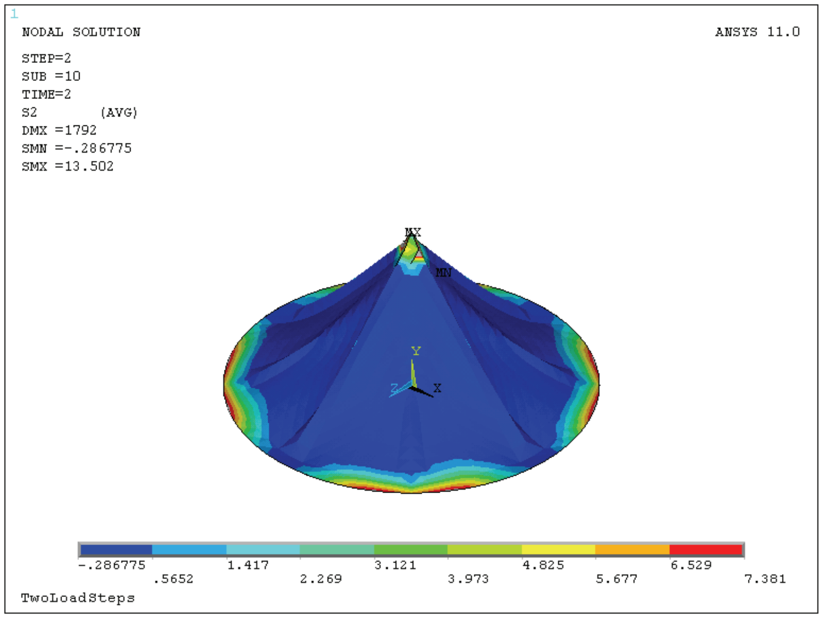


Figure 41 Second Principal Stress Distribution, USERMAT-II, Final Step, [MPa]

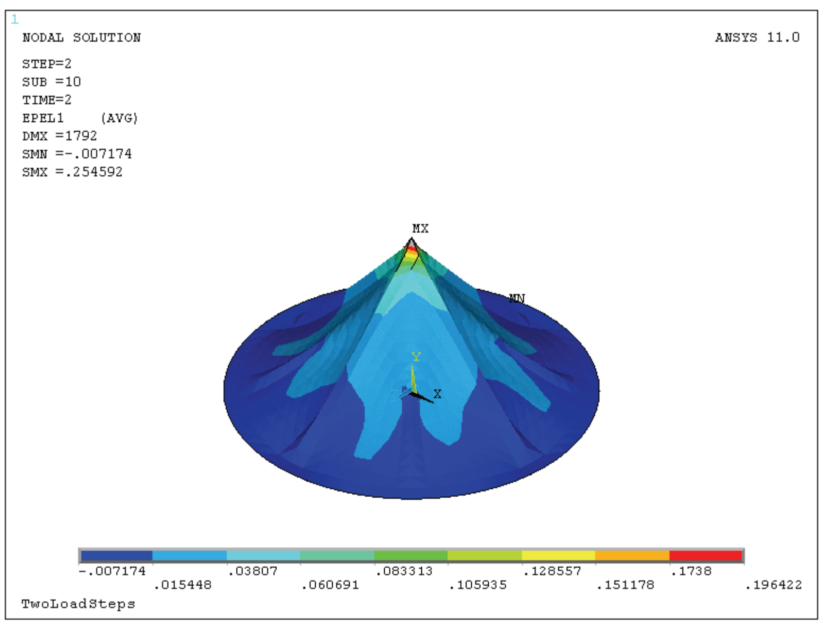


Figure 42 First Principal Strain Distribution, USERMAT-II, Final Step

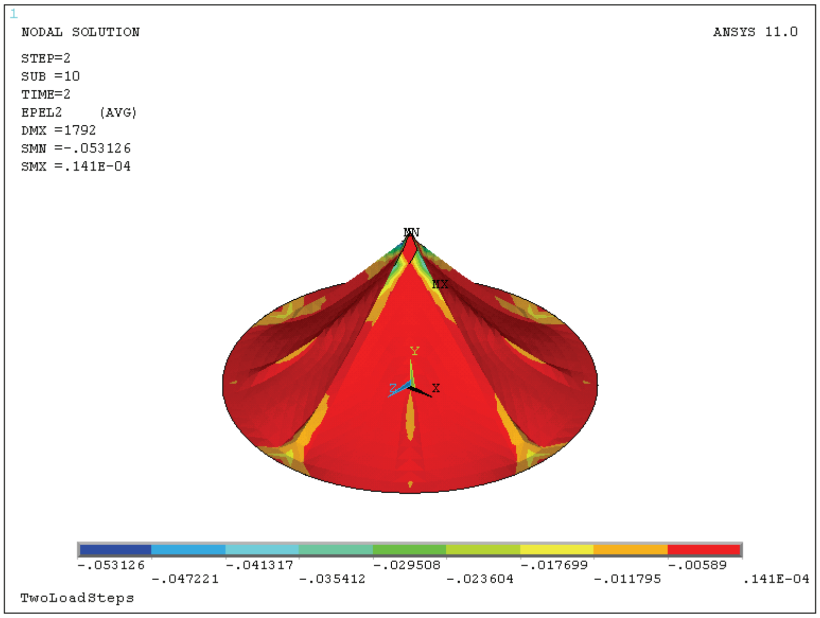


Figure 43 Second Principal Strain Distribution, USERMAT-II, Final Step

6 Conclusions and Recommendations

6.1 Conclusions

- The linear isotropic material model for fabric is limited up to a strain of 1% and therefore is not suitable for the tent structure with a maximum strain around 3%.

- Element from ANSYS 18X family must be chosen in order to implement the USERMAT subroutine. The element SHELL181 is used to make the model. It proves to be suitable for a nonlinear analysis and gives identical results when compared to the ANSYS membrane element SHELL41.

- For an isotropic material model, the prestress caused by temperature is unrealistic and need to be adjusted by a proper factor.

- The thickness of the PTFE coated fiberglass fabric will be defined by ANSYS Real Constants Sets and linked to the SHELL181 element. Therefore it is not necessary to pass the thickness constant to the USERMAT subroutine.

- The compiling and linking of the USERMAT Subroutine can be verified by creating a routine that consists of a linear material model. This user model shows good agreement with the ANSYS material model.

- ANSYS is capable of performing the geometric nonlinear analysis for tent structures using a physically nonlinear fabric model provided by an external USERMAT subroutine.

- Of the three Jacobian Matrices that have been studied, Jacobian-II gives the most efficient solution by using a two-step strain increment.

- The stress-strain relation from an analysis with the nonlinear USERMAT model shows good resemblance with the result from the bi-axial test in Stevin Lab.

- By combining ANSYS script and the USERMAT subroutine, multiple load cases can be defined.

- For the nonlinear model, the prestress can be introduced by using a temperature increment and a prestress coefficient in USERMAT subroutine.

- The automatic time stepping and bisection are able to assist in convergence but cause large calculation times. At the other side a proper fixed step combined with a Map Mesh in ANSYS have shown its efficiency to find the solution. The total calculation time can be reduced within 10 minutes.

6.2 Recommendations

- The element SHELL181 has an option to define initial prestress by using user subroutine USTRESS. This might provide a more accurate way to apply the prestress in the tent structure.

- The Jacobian Matrix calculated by the USERMAT must be symmetric at the current stage. If unsymmetric matrix is desired in a future development, using the PLANE element from 18x family is a possible solution. Because when the element key option KEYOPT(5) is set to be 1, unsymmetric Jacobian matrix is allowed for this type of element.

- The different methods used in this report to obtain an estimation of the exact Jacobian matrix need a further study when less calculation time is desired for a more complex tent structure.

- The effects of automatic time stepping, bisection, fixed step, type of meshing, etc. on convergence matters need a further study for a more complex structural model.

- In the current material model it is assumed that the properties for both warp and weft directions are equivalent. Actually the material constants may vary slightly in different directions. This difference might be added to the USERMAT subroutine to improve the quality of the Model.

- For a good design of a tent structure, the warp and weft direction of the fabric will coincide with the principal direction. Thus the USERMAT model can be improved that it transfers strain and stress between XY directions and principal directions. Efforts have already been spent for this improvement as shown in Figure 44. The USERMAT for this end is included in Appendix V. However, it has not been successfully implemented in an analysis due to convergence problem. Since only eight weeks are allotted to this additional thesis, this problem is left to be solved in a future project.

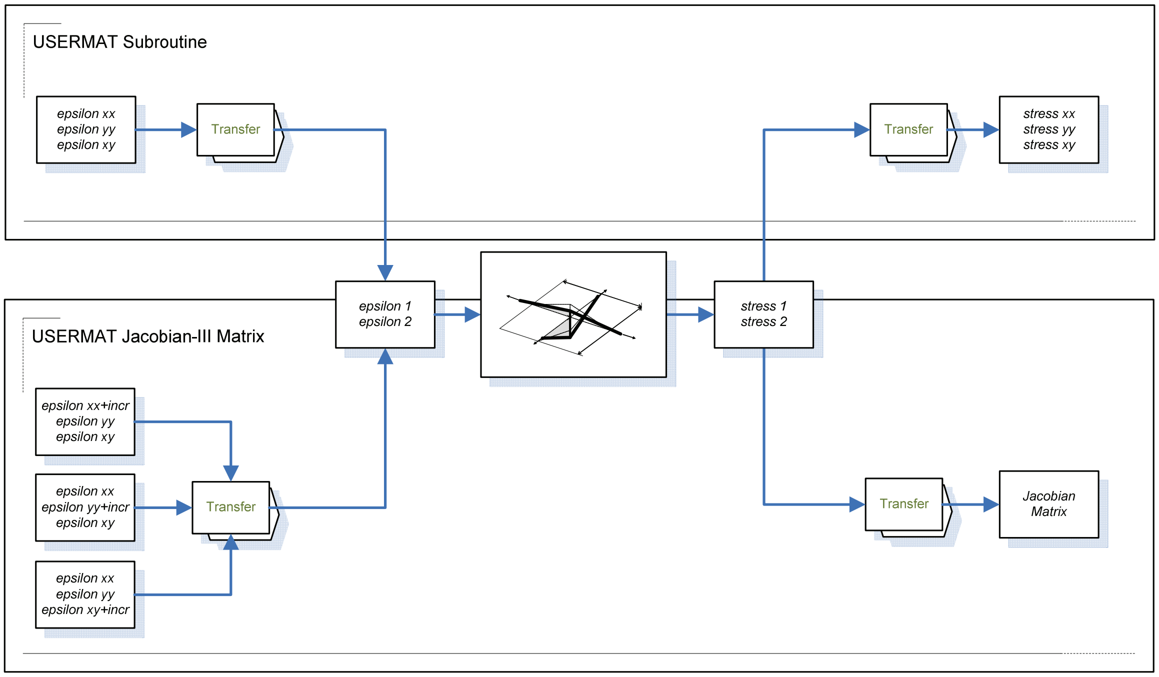


Figure 44 Transfer between X-Y Direction and Principal Direction

Reference

[1] P.C.J. Hoogenboom, Structural model for textile, internal report, Delft University of Technology, Jun. 2007

[2] P.H. van Asselt, Analysis of stressed membrane structures, Master of Science report, Delft University of Technology, Dec. 2007, online: http://www.mechanics.citg.tudelft.nl /~pierre/MSc_projects/reportVanAsselt.pdf

[3] R. Houtman, M. Orpana, Bauen mit Textilen Heft, 4/2000

[4] ANSYS, Inc., ANSYS USER Material Subroutine USERMAT, Nov. 1999

[5] ANSYS, Inc., Guide to ANSYS User Programmable Features, Aug. 2005

APPENDIX-I Tips for USERMAT Linking

- A suitable version of Intel FORTRAN Compiler and C Compiler must be properly installed for the compiling and linking of USERMAT with ANSYS.
- In case of ANSYS 11.0, the Intel FORTRAN 8.1 or a higher version is required. In case of Intel FORTRAN 8.1, Visual Studio 2002 or 2003 is suggested.
- A 30 days trail version of Intel FORTRAN Compiler can be downloaded at www.intel.com/cd/software/products/asmo-na/eng/download/eval/219690.htm.
- A full version of Visual Studio can be downloaded at www.MA3D.com after registration using the E-mail with a domain name of TU Delft.
- If any old version of the compiler has already existed, it is suggested to remove them completely before the installation of a new version.
- The USERMAT source file must be place in the subdirectory [...\AnsysInc\V110\ansys \custom\user\Intel]. It will be linked to ANSYS by select the [Relink ANSYS] option from the ANS_ADMIN utility. A new executable file named [ansys.exe] will be created in the same folder.
- The relinked version of the ANSYS program can be executed by [Configure ANSYS Products].
- If any modification has been made to the USERMAT source file, the [Relink ANSYS] need to be run again. ANSYS will then update the existing [ansys.exe] file. Nevertheless, it is still suggested to delete any existing [ansys.exe] file before running a new relink. This is because sometimes the update may fail, but ANSYS always gives a message that the executable file has been successfully updated.
- If the compilers are not properly installed or the environment variables are not correctly set the linking will fail. In case of latter, some possible error messages, as well as the relevant solutions to solve the problems, are collected in Table 5.
- When using [Relink ANSYS] to compile and link, it is noted that the typing error and the syntax error in the USERMAT will not be reported. In that case it is possible to generate a wrong [ansys.exe] which leads to untrue results without any warning. Therefore the source FORTRAN file should be carefully checked.

- A trick for checking of the typing errors or syntax errors in USERMAT: Create a new folder and copy the following files: USERMAT.f (the source file created by users), ANSCUST.bat, ansyslarge.def, ansyssmall.def and MAKEFILE under the subdirectory [...\AnsysInc\V110\ansys\ custom\user\Intel] and paste them to the new folder. Then by running ANSCUST.bat the USERMAT will be checked and ANSYS will make a report if any error has been detected.

Table 5 Possible Error Messages during Linking Process and the Solutions

Error Message during Relink	Solution [add to Environment Variable]
'nmake' is not recognized as an internal or external command, operable program or batch file.	PATH -> [...\Microsoft Visual Studio .NET 2003\Common7\IDE] and [...\Microsoft Visual Studio .NET 2003\Vc7\bin]
LINK : fatal error LNK1181: cannot open input file 'kernel32.lib'	LIB -> [...\Microsoft Visual Studio .NET 2003\Vc7\lib]
LINK : fatal error LNK1181: cannot open input file 'advapi32.lib'	LIB -> [...\Microsoft Visual Studio .NET 2003\Vc7\PlatformSDK\Lib]
ANSZIP.C(6) : fatal error C1083: Cannot open include file: 'stdio.h': No such file or directory	INCLUDE -> [...\Microsoft Visual Studio .NET 2003\Vc7\include]
ANSZIP.C(10) : fatal error C1083: Cannot open include file: 'windows.h': No such file or directory	INCLUDE -> [...\Microsoft Visual Studio .NET 2003\Vc7\PlatformSDK\Include]
Note: If a different version of Microsoft Visual Studio other than v2003 has been used, the paths listed above need to be changed in accordance with the directory.	

APPENDIX-II USERMAT – Linear Isotropic Model

This USERMAT consists of a linear plain stress algorithm and is written in the purpose of verification of the compiling and linking with ANSYS.

```
*deck,usermat  parallel    user                         gal
     subroutine usermat(
    &               matId, elemId,kDomIntPt, kLayer,
kSectPt,
    &               ldstep,isubst,keycut,
    &               nDirect,nShear,ncomp,nStatev,nProp,
    &               Time,dTime,Temp,dTemp,
    &
stress,ustatev,dsdePl,sedEl,sedPl,epseq,
    &               Strain,dStrain, epsPl, prop, coords,
    &               rotateM, defGrad_t, defGrad,
    &               tsstif, epsZZ,
    &               var1, var2, var3, var4, var5,
    &               var6, var7, var8)
c
#include "impcom.inc"
c
     INTEGER
    &             matId, elemId,
    &             kDomIntPt, kLayer, kSectPt,
    &             ldstep,isubst,keycut,
    &             nDirect,nShear,ncomp,nStatev,nProp
     DOUBLE PRECISION
    &             Time,    dTime,   Temp,    dTemp,
    &             sedEl,   sedPl,   epseq,   epsZZ
     DOUBLE PRECISION
    &             stress  (ncomp  ), ustatev (nStatev),
    &             dsdePl  (ncomp,ncomp),
    &             Strain  (ncomp  ), dStrain (ncomp  ),
    &             epsPl   (ncomp  ), prop    (nProp  ),
    &             coords  (3),       rotateM (3,3),
    &             defGrad (3,3),     defGrad_t(3,3),
    &             tsstif  (2)
c
c***************** User defined part
*************************************
c
c --- parameters
c
     INTEGER         NEWTON, mcomp
     DOUBLE PRECISION HALF, THIRD, ONE, TWO,
THREE, SMALL,
    &             SQTWOTHIRD, SQTWO1,
    &             ZERO, TWOTHIRD, ONEDM02,
ONEDM05, sqTiny
```

```
     PARAMETER       (ZERO       = 0.d0,
    &             HALF       = 0.5d0,
    &             THIRD      = 1.d0/3.d0,
    &             ONE        = 1.d0,
    &             TWO        = 2.d0,
    &             THREE      = 3.d0,
    &             SMALL      = 1.d-08,
    &             sqTiny     = 1.d-20,
    &             ONEDM02    = 1.d-02,
    &             ONEDM05    = 1.d-05,
    &             TWOTHIRD   = 2.0d0/3.0d0,
    &             SQTWOTHIRD =
0.816496580927726030d0,
    &             SQTWO1     =
0.707106769084930420d0,
    &             NEWTON     = 20,
    &             mcomp      = 6
    &             )
c
c --- temperary variables for solution purpose
c
     EXTERNAL         vmove, vzero, vapb1, rotVect
     DOUBLE PRECISION sigElp(mcomp),
dsdeEl(mcomp,mcomp),
    &             wk1(3), wk2(3), wk3(3), wk4(3)

     DOUBLE PRECISION var1, var2, var3, var4, var5,
    &             var6, var7, var8

     INTEGER         i, j, k
     DOUBLE PRECISION pleq_t,  sigy_t , sigy,
    &             dpleq,   pleq,    twoG,    et,
    &             young, posn,    sigy0,   dsigdep, tEo1pm,
    &             gamma, dgamma,  dfdga,   dplga,
    &             funcFb,funcFb2, funcf,   dFdep,   fratio,
    &             con1,  con2,    con3,  con4,
    &             con2p1, ocon2p1,
    &             ocon2p2, con4p1, ocon4p1, ocon4p2,
    &             c1, c2, c3,c4, c5
c*******************************************
******************************
c *** no bsect/cut
     keycut   = 0
c
c *** get Young's modulus and Poisson's ratio
```

```
      young    = prop(1)
      posn     = prop(2)
      twoG     = young / (ONE+posn)
      ncomp    = THREE
c
c *** calculate elastic stiffness matrix (3d)
c
      c1 = ONE - posn * posn
      c2 = young / c1
      c3 = posn * c2
      dsdePl(1,1) = c2
      dsdePl(1,2) = c3
      dsdePl(1,3) = ZERO
      dsdePl(2,2) = c2
      dsdePl(2,3) = ZERO
      dsdePl(3,3) = HALF * twoG
      do i=1,ncomp-1
       do j=i+1,ncomp
        dsdePl(j,i)=dsdePl(i,j)
       end do
      end do
c
c *** calculate elastic strain
      do i=1,ncomp
        wk1(i) = Strain(i) + dStrain(i)
      end do
c
c *** update stresses
      stress(1) = wk1(1) * c2 + wk1(2) * c3
      stress(2) = wk1(1) * c3 + wk1(2) * c2
      stress(3) = wk1(3) * HALF * twoG

      return
           end
```

APPENDIX-III USERMAT-I and APDL Script-I

The USERMAT-I includes the nonlinear fiber interaction model and a Jacobian-II matrix with a two-step strain increment. Prestress is not included. The APDL script-I defines the material constants and applies an upward wind load on the tent structure.

```
*deck,usermat  parallel   user                    gal
    subroutine usermat(
   &                 matId, elemId,kDomIntPt, kLayer,
kSectPt,
   &                 ldstep,isubst,keycut,
   &                 nDirect,nShear,ncomp,nStatev,nProp,
   &                 Time,dTime,Temp,dTemp,
   &
stress,ustatev,dsdePl,sedEl,sedPl,epseq,
   &                 Strain,dStrain, epsPl, prop, coords,
   &                 rotateM, defGrad_t, defGrad,
   &                 tsstif, epsZZ,
   &                 var1, var2, var3, var4, var5,
   &                 var6, var7, var8)
c
#include "impcom.inc"
c
    INTEGER
   &                matId, elemId,
   &                kDomIntPt, kLayer, kSectPt,
   &                ldstep,isubst,keycut,
   &                nDirect,nShear,ncomp,nStatev,nProp
    DOUBLE PRECISION
   &                Time,    dTime,   Temp,    dTemp,
   &                sedEl,   sedPl,   epseq,   epsZZ
    DOUBLE PRECISION
   &                stress  (ncomp  ), ustatev (nStatev),
   &                dsdePl  (ncomp,ncomp),
   &                Strain  (ncomp  ), dStrain (ncomp  ),
   &                epsPl   (ncomp  ), prop    (nProp  ),
   &                coords  (3),       rotateM (3,3),
   &                defGrad (3,3),     defGrad_t(3,3),
   &                tsstif  (2)
c
c**************** User defined part
*************************************
c
c --- parameters
c
    INTEGER         NEWTON, mcomp
    DOUBLE PRECISION HALF, THIRD, ONE, TWO,
THREE, SMALL,
   &                SQTWOTHIRD, SQTWO1, PI, FORTH,
   &                ZERO, TWOTHIRD, ONEDM01,
ONEDM02, ONEDM04, sqTiny

    PARAMETER       (ZERO       = 0.d0,
   &                 HALF       = 0.5d0,
   &                 THIRD      = 1.d0/3.d0,
   &                 FORTH      = 1.d0/4.d0,
   &                 ONE        = 1.d0,
   &                 TWO        = 2.d0,
   &                 THREE      = 3.d0,
   &                 SMALL      = 1.d-08,
   &                 sqTiny     = 1.d-20,
   &                 ONEDM01    = 1.d-01,
   &                 ONEDM02    = 1.d-02,
   &                 ONEDM04    = 1.d-04,
   &                 TWOTHIRD   = 2.0d0/3.0d0,
   &                 SQTWOTHIRD =
0.816496580927726030d0,
   &                 SQTWO1     =
0.707106769084930420d0,
   &                 PI         = 3.1415926d0,
   &                 NEWTON     = 20,
   &                 mcomp      = 6
   &                 )
c
c --- temperary variables for solution purpose
c
    EXTERNAL        vmove, vzero, vapb1, rotVect
    DOUBLE PRECISION sigElp(mcomp),
dsdeEl(mcomp,mcomp),
   &                 wk1(3), wk2(3), wk3(3), wk4(3)

    DOUBLE PRECISION var1, var2, var3, var4, var5,
   &                 var6, var7, var8

    INTEGER          i, j, k
    DOUBLE PRECISION pleq_t,  sigy_t , sigy,
   &                 dpleq,   pleq,   twoG,   et,
   &                 young, posn,   sigy0,  dsigdep, tEo1pm,
   &                 gamma, dgamma,  dfdga,   dplga,
   &                 funcFb,funcFb2, funcf,  dFdep,   fratio,
   &                 con1,  con2,   con3,  con4,
   &                 con2p1, ocon2p1,
   &                 ocon2p2, con4p1, ocon4p1, ocon4p2,
   &                 c1, c2, c3,c4, c5,
   &                 epsxxp, epsyyp, epsxxinc, epsyyinc,
   &                 G, d, s, incr, a, Ad, sx, sy, w, R, ax, ay,
```

```
    &           Tx, Ty, Nx, Ny, nxxp, nyyp, nxyp, nxx,
nyy, nxy
c*******************************************
*****************************
c *** bisect/cut
      keycut   = 0
c
c *** get Young's modulus
      young    = prop(1)
      G        = 500.d0
c
c *** get initial stress, diameter and spacing
      d=prop(2)
      s=prop(3)
      epsxxp=prop(4)
      epsyyp=prop(5)
      incr=ONEDM01
c
c *** calculate elastic strain
      do i=1,ncomp
        wk1(i) = Strain(i) + dStrain(i)
      end do
c
c *** computation
      a=sqrt(s**TWO+d**TWO)
      Ad=FORTH*PI*(d**TWO)
            sx=s*(ONE+wk1(1)+epsxxp)
            sy=s*(ONE+wk1(2)+epsyyp)
            w=ZERO
            R=100.0d0
            do while (abs(R) .gt. ONEDM04*a*d)
              ax=sqrt(sx**TWO+(d-w)**TWO)
              ay=sqrt(sy**TWO+(d+w)**TWO)
              Tx=(ax/a-ONE)
              Ty=(ay/a-ONE)
c
      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif
c
              R=Tx*ay*(d-w)-Ty*ax*(d+w)
              w=w+R/a
            end do

            Tx=young*Ad*Tx
            Ty=young*Ad*Ty
            Nx=(sx*Tx/ax)
            Ny=(sy*Ty/ay)

            nxxp=Nx/s
            nyyp=Ny/s
            nxyp=G*wk1(3)*d
c
c *** add small strain increment in x direction
      epsxxinc=wk1(1)+incr

      a=sqrt(s**TWO+d**TWO)
      Ad=FORTH*PI*(d**TWO)
            sx=s*(ONE+epsxxinc+epsxxp)
            sy=s*(ONE+wk1(2)+epsyyp)
            w=ZERO
            R=100.0d0
            do while (abs(R) .gt. ONEDM04*a*d)
              ax=sqrt(sx**TWO+(d-w)**TWO)
              ay=sqrt(sy**TWO+(d+w)**TWO)
              Tx=(ax/a-ONE)
              Ty=(ay/a-ONE)
c
      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif
c
              R=Tx*ay*(d-w)-Ty*ax*(d+w)
              w=w+R/a
            end do

            Tx=young*Ad*Tx
            Ty=young*Ad*Ty
      Nx=(sx*Tx/ax)
            Ny=(sy*Ty/ay)

      nxx=Nx/s
      nyy=Ny/s

      dsdePl(1,1)=(nxx-nxxp)/incr
      dsdePl(2,1)=(nyy-nyyp)/incr
      dsdePl(3,1)=0
c
c *** add small strain increment in y direction
      epsyyinc=wk1(2)+incr

      a=sqrt(s**TWO+d**TWO)
            Ad=FORTH*PI*(d**TWO)
            sx=s*(ONE+wk1(1)+epsxxp)
            sy=s*(ONE+epsyyinc+epsyyp)
            w=ZERO
            R=100.0d0
            do while (abs(R) .gt. ONEDM04*a*d)
              ax=sqrt(sx**TWO+(d-w)**TWO)
              ay=sqrt(sy**TWO+(d+w)**TWO)
              Tx=(ax/a-ONE)
        Ty=(ay/a-ONE)
c
      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif
c
```

```
  R=Tx*ay*(d-w)-Ty*ax*(d+w)
          w=w+R/a
end do

        Tx=young*Ad*Tx
        Ty=young*Ad*Ty
        Nx=(sx*Tx/ax)
        Ny=(sy*Ty/ay)

nxx=Nx/s
nyy=Ny/s

dsdePl(1,2)=(nxx-nxxp)/incr
dsdePl(2,2)=(nyy-nyyp)/incr
    dsdePl(3,2)=0
    dsdePl(2,1)=dsdePl(1,2)
    dsdePl(1,3)=0
    dsdePl(2,3)=0
    dsdePl(3,3)=G*d
c
c *** update stresses
    stress(1)=nxxp
    stress(2)=nyyp
    stress(3)=nxyp

    return
       end
```

APDL Script-I

```
/filname, UPF, 1
/title, UPF

/UNITS, MPA
/PREP7

D=20000
H=10000

ET, 1, SHELL181
KEYOPT, 1, 1, 0
KEYOPT, 1, 3, 2
R, 1, 1.5

MAT, 1
TB, USER, 1, 1, 5
TBTEMP, 1.0
TBDATA, 1, 51667, 0.18, 0.9, 0, 0
TB, STATE, 1, , 6

K, 1, 0, H, 0
K, 2, D/2, 0, 0
L, 1, 2
K, 1000, 0, 0, 0
AROTAT, 1, , , , , , 1, 1000, 360,

AATT, 1, 1, 1, 0
AESIZE, all, 1000
MSHAPE, 1, 2D
MSHKEY, 1
AMESH, all
FINISH

/SOLU
NSEL, S, LOC, Y, -0.1, 0.1
NSEL, A, NODE, , 0.5, 1.5
D,all,,,,,,all
ALLSEL
SFE, all, 1, pres, , 0.002

NLGEOM, on
SOLCONTROL, on
TIME, 1
NSUBST, 100, 10000, 1, on
AUTOTS, on
KBC, 0
LNSRCH, auto
EQSLV, front
NROPT, full

OUTRES, ALL, 1
OUTPR, ALL, 1
/GST, on
NCNV, 2

ALLSEL
SOLVE
FINISH
```

APPENDIX-IV USERMAT-II and APDL Script-II

This USERMAT-II includes the nonlinear fiber interaction model and a Jacobian-II matrix with two-step strain increments.

```
*deck,usermat  parallel    user                         gal
    subroutine usermat(
   &              matId, elemId,kDomIntPt, kLayer,
kSectPt,
   &              ldstep,isubst,keycut,
   &              nDirect,nShear,ncomp,nStatev,nProp,
   &              Time,dTime,Temp,dTemp,
   &
stress,ustatev,dsdePl,sedEl,sedPl,epseq,
   &              Strain,dStrain, epsPl, prop, coords,
   &              rotateM, defGrad_t, defGrad,
   &              tsstif, epsZZ,
   &              var1, var2, var3, var4, var5,
   &              var6, var7, var8)
c
#include "impcom.inc"
c
    INTEGER
   &             matId, elemId,
   &             kDomIntPt, kLayer, kSectPt,
   &             ldstep,isubst,keycut,
   &             nDirect,nShear,ncomp,nStatev,nProp
    DOUBLE PRECISION
   &             Time,    dTime,   Temp,    dTemp,
   &             sedEl,   sedPl,   epseq,   epsZZ
    DOUBLE PRECISION
   &             stress  (ncomp  ), ustatev (nStatev),
   &             dsdePl  (ncomp,ncomp),
   &             Strain  (ncomp  ), dStrain (ncomp  ),
   &             epsPl   (ncomp  ), prop    (nProp  ),
   &             coords  (3),      rotateM (3,3),
   &             defGrad (3,3),     defGrad_t(3,3),
   &             tsstif  (2)
c
c***************** User defined part
*************************************
c
c --- parameters
c
    INTEGER        NEWTON, mcomp
    DOUBLE PRECISION HALF, THIRD, ONE, TWO,
THREE, SMALL,
   &             SQTWOTHIRD, SQTWO1, PI, FORTH,
   &             ZERO, TWOTHIRD, ONEDM01,
ONEDM02, ONEDM04, sqTiny
    PARAMETER      (ZERO       = 0.d0,
   &             HALF       = 0.5d0,
   &             THIRD      = 1.d0/3.d0,
   &             FORTH      = 1.d0/4.d0,
   &             ONE        = 1.d0,
   &             TWO        = 2.d0,
   &             THREE      = 3.d0,
   &             SMALL      = 1.d-08,
   &             sqTiny     = 1.d-20,
   &             ONEDM01    = 1.d-01,
   &             ONEDM02    = 1.d-02,
   &             ONEDM04    = 1.d-04,
   &             TWOTHIRD   = 2.0d0/3.0d0,
   &             SQTWOTHIRD =
0.816496580927726030d0,
   &             SQTWO1     =
0.707106769084930420d0,
   &             PI         = 3.1415926d0,
   &             NEWTON     = 20,
   &             mcomp      = 6
   &             )
c
c --- temperary variables for solution purpose
c
    EXTERNAL       vmove, vzero, vapb1, rotVect
    DOUBLE PRECISION sigElp(mcomp),
dsdeEl(mcomp,mcomp),
   &             wk1(3), wk2(3), wk3(3), wk4(3)

    DOUBLE PRECISION var1, var2, var3, var4, var5,
   &             var6, var7, var8

    INTEGER        i, j, k
    DOUBLE PRECISION pleq_t,  sigy_t , sigy,
   &             dpleq,   pleq,    twoG,    et,
   &             young, posn,    sigy0,   dsigdep, tEo1pm,
   &             gamma, dgamma,  dfdga,   dplga,
   &             funcFb,funcFb2, funcf,   dFdep,   fratio,
   &             con1,  con2,    con3,  con4,
   &             con2p1, ocon2p1,
   &             ocon2p2, con4p1, ocon4p1, ocon4p2,
   &             c1, c2, c3,c4, c5,
   &             epsxxp, epsyyp, epsxxinc, epsyyinc,
   &             G, d, s, incr, a, Ad, sx, sy, w, R, ax, ay,
   &             Tx, Ty, Nx, Ny, nxxp, nyyp, nxyp, nxx,
nyy, nxy
c*******************************************
*****************************
c *** bisect/cut
```

```
      keycut   = 0
c
c *** get Young's modulus
      young    = prop(1)
      G        = 500.d0
c
c *** get diameter and spacing
      d=prop(2)
      s=prop(3)
c
c *** prestress by temperature
      c1=temp+dtemp
      epsxxp=prop(4)*c1
      epsyyp=prop(5*c1
      incr=ONEDM01
c
c *** calculate elastic strain
      do i=1,ncomp
        wk1(i) = Strain(i) + dStrain(i)
      end do
c
c *** computation
      a=sqrt(s**TWO+d**TWO)
      Ad=FORTH*PI*(d**TWO)
              sx=s*(ONE+wk1(1)+epsxxp)
              sy=s*(ONE+wk1(2)+epsyyp)
              w=ZERO
              R=100.0d0
              do while (abs(R) .gt. ONEDM04*a*d)
                ax=sqrt(sx**TWO+(d-w)**TWO)
                ay=sqrt(sy**TWO+(d+w)**TWO)
                Tx=(ax/a-ONE)
                Ty=(ay/a-ONE)
c
      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif
c
                R=Tx*ay*(d-w)-Ty*ax*(d+w)
                w=w+R/a
              end do

              Tx=young*Ad*Tx
              Ty=young*Ad*Ty
              Nx=(sx*Tx/ax)
              Ny=(sy*Ty/ay)

              nxxp=Nx/s
              nyyp=Ny/s
              nxyp=G*wk1(3)*d
c
c *** add small strain increment in x direction
      epsxxinc=wk1(1)+incr

      a=sqrt(s**TWO+d**TWO)
```

```
      Ad=FORTH*PI*(d**TWO)
              sx=s*(ONE+epsxxinc+epsxxp)
              sy=s*(ONE+wk1(2)+epsyyp)
              w=ZERO
              R=100.0d0
              do while (abs(R) .gt. ONEDM04*a*d)
                ax=sqrt(sx**TWO+(d-w)**TWO)
                ay=sqrt(sy**TWO+(d+w)**TWO)
                Tx=(ax/a-ONE)
                Ty=(ay/a-ONE)
c
      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif
c
                R=Tx*ay*(d-w)-Ty*ax*(d+w)
                w=w+R/a
              end do

              Tx=young*Ad*Tx
              Ty=young*Ad*Ty
      Nx=(sx*Tx/ax)
              Ny=(sy*Ty/ay)

      nxx=Nx/s
      nyy=Ny/s

      dsdePl(1,1)=(nxx-nxxp)/incr
      dsdePl(2,1)=(nyy-nyyp)/incr
      dsdePl(3,1)=0
c
c *** add small strain increment in y direction
      epsyyinc=wk1(2)+incr

      a=sqrt(s**TWO+d**TWO)
              Ad=FORTH*PI*(d**TWO)
              sx=s*(ONE+wk1(1)+epsxxp)
              sy=s*(ONE+epsyyinc+epsyyp)
              w=ZERO
              R=100.0d0
              do while (abs(R) .gt. ONEDM04*a*d)
                ax=sqrt(sx**TWO+(d-w)**TWO)
                ay=sqrt(sy**TWO+(d+w)**TWO)
                Tx=(ax/a-ONE)
        Ty=(ay/a-ONE)
c
      if (Tx .LT. 0) then
        Tx=0
      endif
      if (Ty .LT. 0) then
        Ty=0
      endif
c
        R=Tx*ay*(d-w)-Ty*ax*(d+w)
                w=w+R/a
```

```
end do

        Tx=young*Ad*Tx
        Ty=young*Ad*Ty
        Nx=(sx*Tx/ax)
        Ny=(sy*Ty/ay)

nxx=Nx/s
nyy=Ny/s

dsdePl(1,2)=(nxx-nxxp)/incr
dsdePl(2,2)=(nyy-nyyp)/incr
dsdePl(3,2)=0
    dsdePl(2,1)=dsdePl(1,2)
    dsdePl(1,3)=0
    dsdePl(2,3)=0
    dsdePl(3,3)=G*d
c
c *** update stresses
    stress(1)=nxxp
    stress(2)=nyyp
    stress(3)=nxyp

    return
          end
```

APDL Script-II

```
/filname, TwoLoadSteps, 1
/title, TwoLoadSteps

/UNITS, MPA
/PREP7

D=20000
H=10000

ET, 1, SHELL181
KEYOPT, 1, 1, 0
KEYOPT, 1, 3, 2
R, 1, 1.5

MAT, 1
TB, USER, 1, 1, 5
!TBTEMP, 1.0
TBDATA, 1, 51667, 0.18, 0.9, 0.001, 0.001
TB, STATE, 1, , 6

K, 1, 0, H, 0
K, 2, D/2, 0, 0
L, 1, 2
K, 1000, 0, 0, 0
AROTAT, 1, , , , , , 1, 1000, 360

AATT, 1, 1, 1, 0
AESIZE, all, 1000

MSHAPE, 1, 2D
MSHKEY, 1
AMESH, all
FINISH

/SOLU
NSEL, S, LOC, Y, -0.1, 0.1
NSEL, A, NODE, , 0.5, 1.5
D,all,,,,,,all

! Load Step 1:

ALLSEL
TREF, 0
BFE, all, temp, 1, 10

NLGEOM, on
SOLCONTROL, on
TIME, 1
NSUBST, 10
AUTOTS, off
KBC, 0
LNSRCH, auto
EQSLV, front
NROPT, full
NEQIT, 2000

OUTRES, ALL, 1
OUTPR, ALL, 1
/GST, on
NCNV, 0

LSWRITE

! Load Step 2:

ALLSEL
SFE, all, 1, pres, , -0.001

NLGEOM, on
SOLCONTROL, on
TIME, 2
NSUBST, 10
AUTOTS, off
KBC, 0
LNSRCH, auto
EQSLV, front
```

```
NROPT, full
NEQIT, 2000

OUTRES, ALL, 1
OUTPR, ALL, 1
/GST, on

NCNV, 0

LSWRITE

LSSOLVE, 1, 2
FINISH
```

APPENDIX-V USERMAT - Transfer between XY and Principal Direction

```
*deck,usermat  parallel   user                    gal
    subroutine usermat(
   &             matId, elemId,kDomIntPt, kLayer,
kSectPt,
   &             ldstep,isubst,keycut,
   &             nDirect,nShear,ncomp,nStatev,nProp,
   &             Time,dTime,Temp,dTemp,
   &
stress,ustatev,dsdePl,sedEl,sedPl,epseq,
   &             Strain,dStrain, epsPl, prop, coords,
   &             rotateM, defGrad_t, defGrad,
   &             tsstif, epsZZ,
   &             var1, var2, var3, var4, var5,
   &             var6, var7, var8)
c
#include "impcom.inc"
c
    INTEGER
   &             matId, elemId,
   &             kDomIntPt, kLayer, kSectPt,
   &             ldstep,isubst,keycut,
   &             nDirect,nShear,ncomp,nStatev,nProp
    DOUBLE PRECISION
   &             Time,   dTime,  Temp,   dTemp,
   &             sedEl,  sedPl,  epseq,  epsZZ
    DOUBLE PRECISION
   &             stress  (ncomp  ), ustatev (nStatev),
   &             dsdePl  (ncomp,ncomp),
   &             Strain  (ncomp  ), dStrain (ncomp  ),
   &             epsPl   (ncomp  ), prop    (nProp  ),
   &             coords  (3),      rotateM (3,3),
   &             defGrad (3,3),    defGrad_t(3,3),
   &             tsstif  (2)
c
c**************** User defined part
*************************************
c
c --- parameters
c
    INTEGER         NEWTON, mcomp
    DOUBLE PRECISION HALF, THIRD, ONE, TWO,
THREE, SMALL,
   &             SQTWOTHIRD, SQTWO1, PI, FORTH,
   &             ZERO, TWOTHIRD, ONEDM01,
ONEDM02, ONEDM04, sqTiny
    PARAMETER      (ZERO       = 0.d0,
   &             HALF       = 0.5d0,
   &             THIRD      = 1.d0/3.d0,
   &             FORTH      = 1.d0/4.d0,
   &             ONE        = 1.d0,
   &             TWO        = 2.d0,
   &             THREE      = 3.d0,
   &             SMALL      = 1.d-08,
   &             sqTiny     = 1.d-20,
   &             ONEDM01    = 1.d-01,
   &             ONEDM02    = 1.d-02,
   &             ONEDM04    = 1.d-04,
   &             TWOTHIRD   = 2.0d0/3.0d0,
   &             SQTWOTHIRD =
0.816496580927726030d0,
   &             SQTWO1     =
0.707106769084930420d0,
   &             PI         = 3.1415926d0,
   &             NEWTON     = 20,
   &             mcomp      = 6
   &             )
c
c --- temperary variables for solution purpose
c
    EXTERNAL        vmove, vzero, vapb1, rotVect
    DOUBLE PRECISION sigElp(mcomp),
dsdeEl(mcomp,mcomp),
   &             wk1(3), wk2(3), wk3(3), wk4(3)

    DOUBLE PRECISION var1, var2, var3, var4, var5,
   &             var6, var7, var8

    INTEGER         i, j, k
    DOUBLE PRECISION pleq_t,  sigy_t , sigy,
   &             dpleq,   pleq,    twoG,    et,
   &             young, posn,    sigy0,   dsigdep, tEo1pm,
   &             gamma, dgamma,  dfdga,   dplga,
   &             funcFb,funcFb2, funcf,   dFdep,   fratio,
   &             con1,  con2,    con3,  con4,
   &             con2p1, ocon2p1,
   &             ocon2p2, con4p1, ocon4p1, ocon4p2,
   &             c1, c2, c3,c4, c5,
   &             epsxxp, epsyyp, epsxxinc, epsyyinc,
epsxyinc,
   &             G, d, s, incr, a, Ad, sx, sy, w, R, ax, ay,
   &             TX, Ty, Nx,Ny, nxxp, nyyp, nxyp, nxx, nyy,
nxy,
   &             aa, bb, rr, cc, dd, sine, cosine,
   &             nxxx, nyyy, nxyxy, nxxinc, nyyinc, nxyinc,
   &             jacobian(3,3)
c
c*******************************************
*****************************
c *** bisect/cut
    keycut   = 0
c
c *** get Young's modulus
```

```
      young    = prop(1)
      G        = 500.d0
c     ncomp    = THREE
c
c *** get diameter and spacing
      d=prop(2)
      s=prop(3)
      epsxxp=ZERO
      epsyyp=ZERO
      incr=ONEDM01
c
c *** calculate elastic strain in x-y direction
      do i=1,ncomp
        wk2(i) = Strain(i) + dStrain(i)
      end do
c
c *** transfer x-y strain to principal strain
      aa=wk2(2)+wk2(1)
      bb=wk2(2)-wk2(1)
      rr=sqrt(bb**TWO+wk2(3)**TWO)
      wk1(1)=HALF*(aa+rr)
      wk1(2)=HALF*(aa-rr)
      if (rr .EQ. 0) then
        cosine=ONE
        sine=ZERO
      else
        cosine=bb/rr
        sine=wk2(3)/rr
      endif
c
c *** computation
      a=sqrt(s**TWO+d**TWO)
      Ad=FORTH*PI*(d**TWO)
              sx=s*(ONE+wk1(1)+epsxxp)
              sy=s*(ONE+wk1(2)+epsyyp)
              w=ZERO
              R=100.0d0
              do while (abs(R) .GT. ONEDM04*a*d)
                ax=sqrt(sx**TWO+(d-w)**TWO)
                ay=sqrt(sy**TWO+(d+w)**TWO)
                Tx=(ax/a-ONE)
                Ty=(ay/a-ONE)
                R=Tx*ay*(d-w)-Ty*ax*(d+w)
                w=w+R/a
              end do

              Tx=young*Ad*Tx
              Ty=young*Ad*Ty
              Nx=(sx*Tx/ax)
              Ny=(sy*Ty/ay)

              nxxp=Nx/s
              nyyp=Ny/s
c
c *** transfer principal stress to x-y stress
      cc=HALF*(nxxp+nyyp)
      dd=HALF*(nxxp-nyyp)
      nxxx=cc-dd*cosine
      nyyy=cc+dd*cosine
      nxyxy=dd*sine
c
c *** update stresses
      stress(1)=nxxx
      stress(2)=nyyy
      stress(3)=nxyxy
c
c *** add small strain increment in x direction
      epsxxinc=wk2(1)+incr
c
c *** transfer x-y strain to principal strain
      aa=wk2(2)+epsxxinc
      bb=wk2(2)-epsxxinc
      rr=sqrt(bb**TWO+wk2(3)**TWO)
      wk1(1)=HALF*(aa+rr)
      wk1(2)=HALF*(aa-rr)
      if (rr .EQ. 0) then
        cosine=ONE
        sine=ZERO
      else
        cosine=bb/rr
        sine=wk2(3)/rr
      endif
c
c *** calculate principal stress
      a=sqrt(s**TWO+d**TWO)
      Ad=FORTH*PI*(d**TWO)
              sx=s*(ONE+wk1(1)+epsxxp)
              sy=s*(ONE+wk1(2)+epsyyp)
              w=ZERO
              R=100.0d0
              do while (abs(R) .GT. ONEDM04*a*d)
                ax=sqrt(sx**TWO+(d-w)**TWO)
                ay=sqrt(sy**TWO+(d+w)**TWO)
                Tx=(ax/a-ONE)
                Ty=(ay/a-ONE)
                R=Tx*ay*(d-w)-Ty*ax*(d+w)
                w=w+R/a
              end do

              Tx=young*Ad*Tx
              Ty=young*Ad*Ty
      Nx=(sx*Tx/ax)
              Ny=(sy*Ty/ay)

      nxx=Nx/s
      nyy=Ny/s
c
c *** transfer principal stress to x-y stress
      cc=HALF*(nxx+nyy)
      dd=HALF*(nxx-nyy)
      nxxinc=cc-dd*cosine
      nyyinc=cc+dd*cosine
      nxyinc=dd*sine

      jacobian(1,1)=(nxxinc-nxxx)/incr
      jacobian(2,1)=(nyyinc-nyyy)/incr
```

```
     jacobian(3,1)=(nxyinc-nxyxy)/incr
c
c *** add small strain increment in y direction
     epsyyinc=wk2(2)+incr
c
c *** transfer x-y strain to principal strain
     aa=epsyyinc+wk2(1)
     bb=epsyyinc-wk2(1)
     rr=sqrt(bb**TWO+wk2(3)**TWO)
     wk1(1)=HALF*(aa+rr)
     wk1(2)=HALF*(aa-rr)
     if (rr .EQ. 0) then
       cosine=ONE
       sine=ZERO
     else
       cosine=bb/rr
       sine=wk2(3)/rr
     endif
c
c *** calculate principal stress
     a=sqrt(s**TWO+d**TWO)
           Ad=FORTH*PI*(d**TWO)
           sx=s*(ONE+wk1(1)+epsxxp)
           sy=s*(ONE+wk1(2)+epsyyp)
           w=ZERO
           R=100.0d0
           do while (abs(R) .GT. ONEDM04*a*d)
             ax=sqrt(sx**TWO+(d-w)**TWO)
             ay=sqrt(sy**TWO+(d+w)**TWO)
             Tx=(ax/a-ONE)
       Ty=(ay/a-ONE)
       R=Tx*ay*(d-w)-Ty*ax*(d+w)
             w=w+R/a
     end do

           Tx=young*Ad*Tx
           Ty=young*Ad*Ty
           Nx=(sx*Tx/ax)
           Ny=(sy*Ty/ay)

     nxx=Nx/s
     nyy=Ny/s
c
c *** transfer principal stress to x-y stress
     cc=HALF*(nxx+nyy)
     dd=HALF*(nxx-nyy)
     nxxinc=cc-dd*cosine
     nyyinc=cc+dd*cosine
     nxyinc=dd*sine

     jacobian(1,2)=((nxxinc-nxxx)/incr +
jacobian(2,1))/TWO
     jacobian(2,1)=jacobian(1,2)
     jacobian(2,2)=(nyyinc-nyyy)/incr
     jacobian(3,2)=(nxyinc-nxyxy)/incr
c *** add small shear strain
     epsxyinc=wk2(3)+incr
c *** transfer x-y strain to principal strain
     aa=wk2(2)+wk2(1)
     bb=wk2(2)-wk2(1)
     rr=sqrt(bb**TWO+epsxyinc**TWO)
     wk1(1)=HALF*(aa+rr)
     wk1(2)=HALF*(aa-rr)
     if (rr .EQ. 0) then
       cosine=ONE
       sine=ZERO
     else
       cosine=bb/rr
       sine=epsxyinc/rr
     endif
c *** calculate principal stress
     a=sqrt(s**TWO+d**TWO)
           Ad=FORTH*PI*(d**TWO)
           sx=s*(ONE+wk1(1)+epsxxp)
           sy=s*(ONE+wk1(2)+epsyyp)
           w=ZERO
           R=100.0d0
           do while (abs(R) .GT. ONEDM04*a*d)
             ax=sqrt(sx**TWO+(d-w)**TWO)
             ay=sqrt(sy**TWO+(d+w)**TWO)
             Tx=(ax/a-ONE)
       Ty=(ay/a-ONE)
       R=Tx*ay*(d-w)-Ty*ax*(d+w)
             w=w+R/a
     end do
           Tx=young*Ad*Tx
           Ty=young*Ad*Ty
           Nx=(sx*Tx/ax)
           Ny=(sy*Ty/ay)

     nxx=Nx/s
     nyy=Ny/s
c *** transfer principal stress to x-y stress
     cc=HALF*(nxx+nyy)
     dd=HALF*(nxx-nyy)
     nxxinc=cc-dd*cosine
     nyyinc=cc+dd*cosine
     nxyinc=dd*sine

     jacobian(1,3)=((nxxinc-
nxxx)/incr+jacobian(3,1))/TWO
     jacobian(3,1)=jacobian(1,3)
     jacobian(2,3)=((nyyinc-
nyyy)/incr+jacobian(3,2))/TWO
     jacobian(3,2)=jacobian(2,3)
     jacobian(3,3)=(nxyinc-nxyxy)/incr

     do i=1,3
       do j=1,3
         dsdeP1(i,j)=jacobian(i,j)
       end do
     end do

     return
        end
```

APPENDIX-VI CD ROM Contents

A CD Rom contains the following information is included:

- This report in PDF
- USERMAT subroutines
- ANSYS APDL scripts
- Results from the analysis
- References and articles