Lehrstuhl für Angewandte Mechanik
Technische Universität München

Yifei Wang

Expand of Finite Element Toolbox

Master's Thesis

30. 05. 2016

Supervisor:
Johannes Rutzmoser

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor Johannes Rutzmoser for the continuous support of my study and related research, for his patience, motivation, and immense knowledge I am deeply grateful of his help in the completion of this thesis. His guidance helped me in all the time of finite element research and get in with Finite- Element- Research-code. I could not have imagined having a better advisor and mentor for my study in Finite-Element- Method.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The finite element method (FEM) is a numerical technique for finding approximate solutions to boundary value problems for partial differential equations. FEM contains linear FEM and non-linear FEM. Linear analysis follows the equation $K \cdot D = F$, in which K stands for stiffness matrix, D stands for displacement, and F stands for force. It means that the correlation of force and displacement is linear. This relation between force and displacement is only valid for materials that have elastic linear property. But in real situation, elastic materials such as steel has also non- linear region. (see the Figure 1.1). The property of steel is elastic linear up until yield point. Then the steel is yielding and becomes non-linear. Non-linear states include geometry non-linear and material non-linear. The example in Figure 1.1 describes a non-linear case and it also the main topic of this thesis. In the fields of engineering and science, FEM is a powerful tool in producing strength visualization and can help engineers to minimize weight, materials and costs. The accuracy of solution are only limited by the quality of model and by the available computational power. Ever since the computational power has been improved enormously, the FEM software has offered a wide range of simulations of complex model designs and system analyses.
AMfe is a nonlinear finite element code for structural application at the chair of applied mechanics in Technical University of Munich. AMfe Toolbox is developed in Python and Fortran. Python is a high-level, interpreted, and dynamically-typed programming language. There are many numerical packages built on Python, such as Numpy, Scipy, Pandas. These packages provide high-performance and easy to use data structures, both of which match the FEM developing work. The fact that Python is a high-level programming language makes it less time-consuming to develop the code. However, Python is slow for repeated execution of low-level task. Each Python operation comes with a small type-checking overhead, and when many repeated small operations add up, the overhead becomes significant. For this reason, A part of the AMfe code is rewritten in Fortran. Fortran is a general-purpose and imperative programming language that is specially tailored for numeric computation and scientific computing. Therefore, the combination of Python and Fortran retains the advantages of both - the easy-to-develop nature of Python and the fast numeric computation in Fortran. The aim of the AMfe Toolbox is to solve and anal-

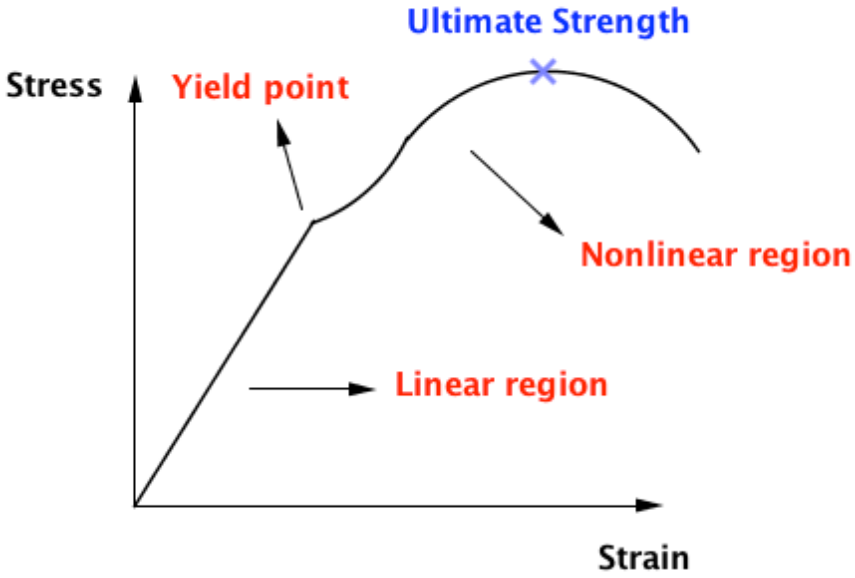


Figure 1.1: Stress-strain curve for steel

yse FEM problems, especially structural mechanical problems. The toolbox contains several modules with different functions to solve problems step by step. A simple structure of AMfe is depicted in Figure 1.3. Displacement, stress, and strain are three important factors in structural mechanical engineering. The function for calculating nodal displacements is completed in AMfe Toolbox, but functions for calculating stress and strain remain blank. Strain and stress calculations are important in structural mechanical analysis and geometric design. For this reason, functions for solving of stress and strain will be added to AMfe Toolbox. When strain and stress calculations are completed, the aim of research is then shifted to improving the accuracy of these results. In order to do so, stress recovery is an approach to extrapolate the element solution to nodal solution. The goal is to be as accurate as possible in the computed displacements while keeping the computational effort reasonable. When the most important functions are completed and running well, it is time to check if the AMfe Toolbox exports the reasonable results. The results of stress and strain from AMfe Toolbox will be compared with a commercial FEM software - ANSYS. The comparison of each element type will be recorded and discussed.

1.2 Structure of Study

This study can be divided into three parts. Figure 1.2 shows the structural flow chart in this study. The first part is marked with a green box. The main function for this part is to export the data of a meshed geometry using ANSYS Parametric Design Language(APDL) code and import the

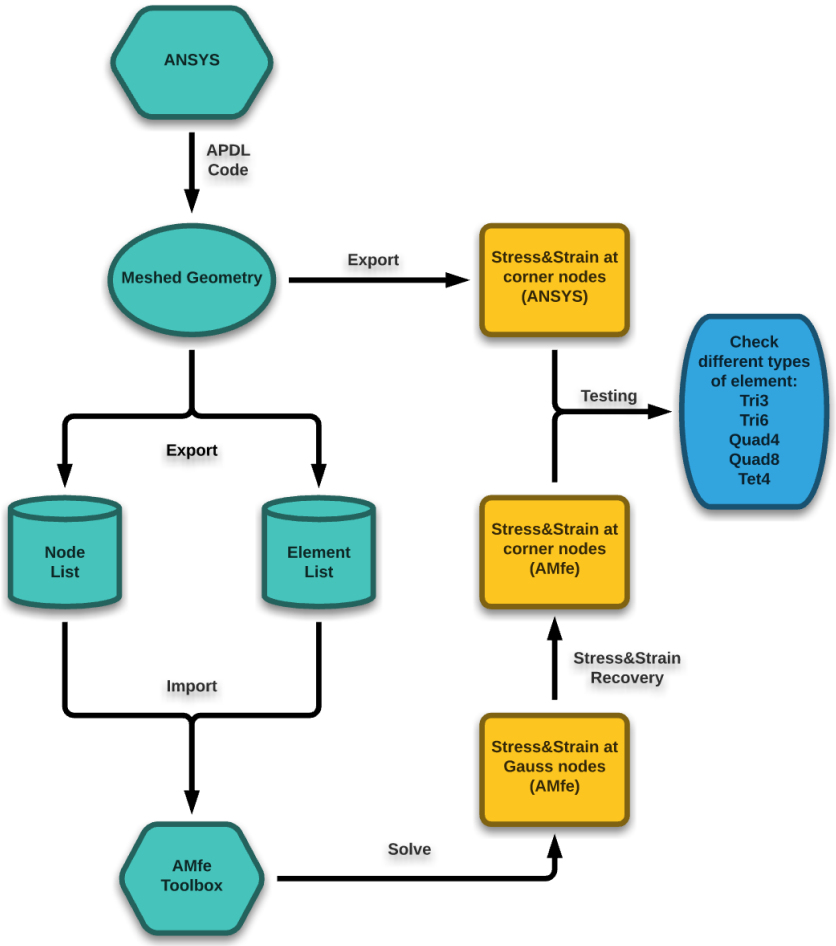


Figure 1.2: Flow chart of structure

data into the AMfe Toolbox. This procedure, which is called Pre-Processing, is the first step in solving a problem in Finite Element Analysis. This step also ensures the analyses from both ANSYS and AMfe Toolbox have the same set of elements and nodes. The processing step comes next, which is marked with a yellow box. AMfe Toolbox has several modules with different functions. These modules combined provides all calculation functions needed for processing. This study is focusing on calculating stress and strain. The last part marked by blue is to check the accuracy of strain and stress using different types of elements.

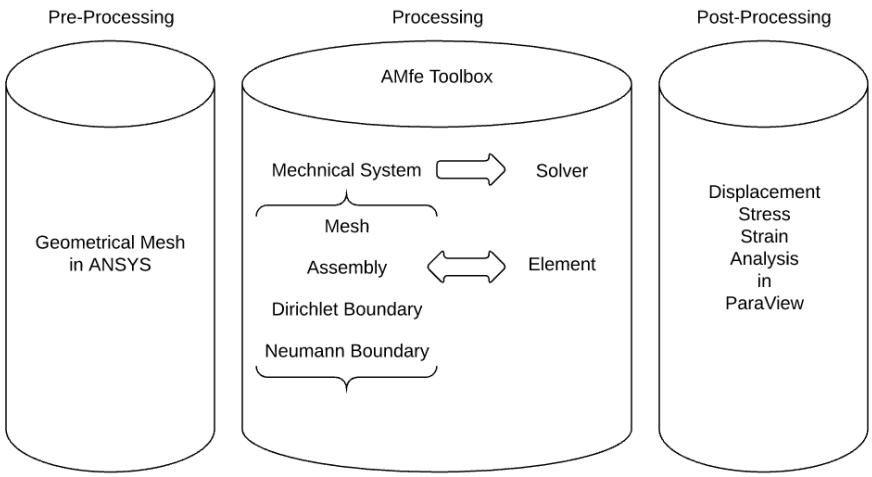


Figure 1.3: Overview of AMfe structure

Chapter 2

Numerical Aspect for Finite Element Formulation

2.1 Shape Function

2.1.1 Quadrilateral Elements

The goal of the AMfe Code in the structural mechanical field is to get the approximate solutions to position, displacement, stress, and strain. The approximation is dependent on the shape function N. The standard approach to define the shape functions is choosing them as simple polynomials that are associated to nodes. In one element, there are n nodes and n shape functions, in which the i-th shape function takes the value of 1 at the i-th node in reference coordinates, then the j-th shape function can be written like this:

$$\begin{aligned} &\text{if } j = i \text{ then } N_j\left(\hat{\xi}_i\right) = 1 \\ &\text{if } j \neq i \text{ then } N_j\left(\hat{\xi}_i\right) = 0 \end{aligned}$$

for i,j = 1,...,n.

In Figure 2.1, linear and quadratic shape functions are shown together with corresponding node positions. When referring to an element with three nodes, as shown in the right of Figure 2.1. For each shape functions, there are three restrictions:

$$\begin{aligned} &N_1\left(\hat{\xi}_1 = -1\right) = 1,\ N_1\left(\hat{\xi}_2 = 0\right) = 0,\ N_1\left(\hat{\xi}_3 = 1\right) = 0; \\ &N_2\left(\hat{\xi}_1 = -1\right) = 0,\ N_2\left(\hat{\xi}_2 = 0\right) = 1,\ N_2\left(\hat{\xi}_3 = 1\right) = 0; \\ &N_3\left(\hat{\xi}_1 = -1\right) = 0,\ N_3\left(\hat{\xi}_2 = 0\right) = 0,\ N_3\left(\hat{\xi}_3 = 1\right) = 1 \end{aligned}$$

For quadratic polynomial, the shape function has three coefficients, and it can be constructed as follows:

$$p_{quad}\left(\xi\right) = c_0 + c_1\xi + c_2\xi^2$$

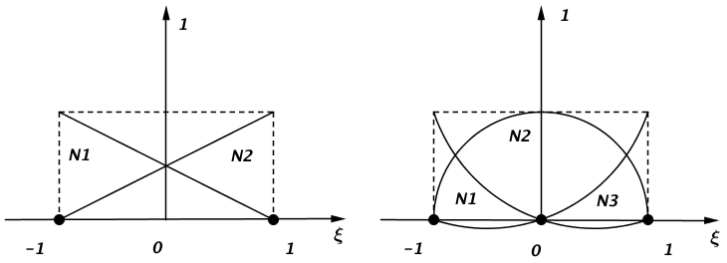


Figure 2.1: Linear shape function and quadratic shape function.

Lagrange polynomials are a general group of polynomials with the node undermentioned association property: a Lagrange polynomial $l_k^n - 1$ (in one coordinate ξ) of order $n-1$ passes through n nodes with coordinates $\bar{\xi}^j$ $(j = 1, ..., n)$ of which the single node k equals one $\left(l_k^{n-1}\left(\bar{\xi}^k\right) = 1\right)$ and every other node results in zero $\left(l_k^{n-1}\left(\bar{\xi}^k\right) = 0 \quad \text{for all} \quad j \neq k\right)$.

$$l_k^{n-1}(\xi) = \prod_{j=1, j\neq k}^{n} \frac{\xi - \bar{\xi}^1}{\bar{\xi}^k - \bar{\xi}^j} = \frac{(\xi-\bar{\xi}^1)\cdots(\xi-\bar{\xi}^{k-1})(\xi-\bar{\xi}^{k+1})\cdots(\xi-\bar{\xi}^n)}{(\bar{\xi}^k-\bar{\xi}^1)\cdots(\bar{\xi}^k-\bar{\xi}^{k-1})(\bar{\xi}^k-\bar{\xi}^{k+1})\cdots(\bar{\xi}^k-\bar{\xi}^n)}$$

$$k = 1, 2...., n$$

For instance, to calculate linear, i.e. order 1, Lagrange polynomials can be calculated by $n-1 = 1 \Rightarrow n = 2$. Using $\bar{\xi}^1 = -1$ and $\bar{\xi}^2 = +1$ results in: $l_1^1(\xi) = -1/2(\xi - 1)$ and $l_2^1(\xi) = 1/2(\xi + 1)$. Then the linear shape functions can be simply constructed as $N_1 = l_1^1 = 1/2(1 - \xi)$ and $N_2 = l_2^1 = 1/2(1 + \xi)$.

2.1.2 Triangular and Tetrahedral Elements

Triangular and tetrahedral elements have higher flexibility than that of quadrilateral and hexahedral elements in meshing complex geometries, thus the former is often preferred to the latter. The shape functions can be solved by an analogous method, but Triangular and tetrahedral elements are expressed in area and volume coordinates, respectively. Figure 2.2 shows the geometries and shape functions. These coordinates represent area and volume fractions, and visualization is given below:

$$L_1 = \frac{\text{area}P23}{\text{area}123}, \qquad V_1 = \frac{\text{area}P234}{\text{area}1234}$$

$$\sum L_i = 1 \qquad \sum V_i = 1$$

2.2 Non-linear Element Formulation

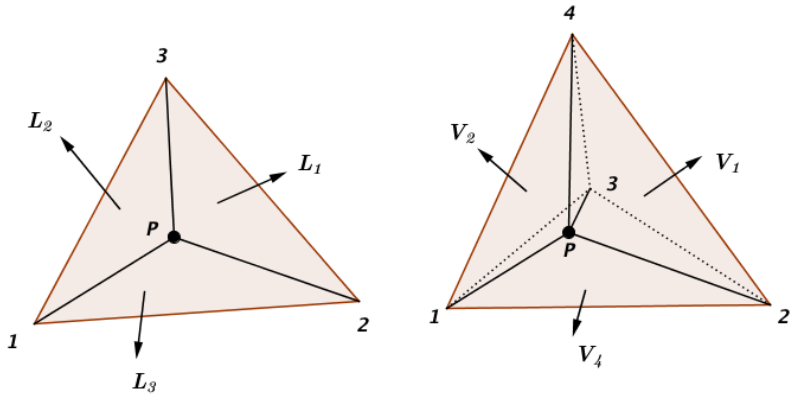


Figure 2.2: triangular and tetradral elemnets.

2.2.1 Discretization of the Displacement Field

After meshing the geometry, We discretize displacement field u by inserting shape function N. Firstly, we introduce the nodal displacement d^e to record displacements at each node of the considered element in each spatial direction. In elastic materials, It can be derived through the elastic constitutive equations that stress σ are directly related to strain ε at each point:

$$\sigma = E\varepsilon \tag{2.1}$$

It follows that the stress calculation procedure begins with strain calculation, and that the accuracy of stresses depends on that of strains. In the following sections, we focus our attention on two-dimensional isoparametric elements, as the calculation of strains, stresses and axial forces in bar elements is strightforward. Suppose that we have solved the master stiffness equations.

$$Ku = f \tag{2.2}$$

for the node displacements u. To calculate strains and stresses we perform a loop over all defined elements. Let ε be the element index of a specific two-dimensional isoparametric element encoun- tered during this loop, and $u(\varepsilon)$ the vector of computed element node displacements. The strains at any point in the element can be related to these displacements as

$$\varepsilon = Bu(\varepsilon) \tag{2.3}$$

where B is the strain-displacement matrix assembled with the x and y derivatives of the element shape functions evaluated at the point where we are calculating strains. The corresponding stresses are given by

$$\sigma = E\varepsilon = EBu \tag{2.4}$$

In the applications it is of interest to evaluate and report these stresses at the element nodal points located on the corners and possibly midpoints of the element. These are called element nodal

point stresses. It is important to realize that the stresses computed at the same nodal point from adjacent elements will not generally be the same, since stresses are not required to be continuous in displacement- assumed finite elements. This suggests some form of stress averaging can be used to improve the stress accuracy, and indeed this is part of the stress recovery technique further. The results from this averaging procedure are called nodal point stresses.

2.3 Non-linear Formulation of Strain and Stress

This section provides an introduction to the most common model and simulation technique for both 2D and 3D solid bodies in Non-Finite-Element-Method. The following section is based on the work of Johaness Rutzmoser[Rutzmoser 2016] in non-linear formulation of strain and stress. In the first part of this introduction, we will pick a fictional 3D element with five nodes as our object of study. The coordinate system is based on three coordinate, namely ξ_1, ξ_2 ,and ξ_3. The shape function of this 3D element can be ordered in Voigt notation as follows:

$$\boldsymbol{N}(\xi) = \begin{pmatrix} N_1(\xi) \\ N_2(\xi) \\ N_3(\xi) \\ N_4(\xi) \\ N_5(\xi) \end{pmatrix} \tag{2.5}$$

We can denote the coordinate of this element as vector $\boldsymbol{X}^e$. The columns of $\boldsymbol{X}^e$ stands for its axis direction and the rows of $\boldsymbol{X}^e$ stands for the index of nodes. The matrix of $\boldsymbol{X}^e$ is shown as:

$$\boldsymbol{X}^e = \begin{pmatrix} X_1 & Y_1 & Z_1 \\ X_2 & Y_2 & Z_2 \\ X_3 & Y_3 & Z_3 \\ X_4 & Y_4 & Z_5 \\ X_5 & Y_5 & Z_6 \end{pmatrix} \tag{2.6}$$

The coordinates $\boldsymbol{X}$ of this element in the initial configuration relies on the local coordinate of the element ξ and the shape function $\boldsymbol{N}$. It can be described as:

$$\boldsymbol{X}(\xi) = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \left(\boldsymbol{X}^e\right)^T N = \begin{pmatrix} X_1 & X_2 & X_3 & X_4 & X_5 \\ Y_1 & Y_2 & Y_3 & Y_4 & Y_5 \\ Z_1 & Z_2 & Z_3 & Z_4 & Z_5 \end{pmatrix} \begin{pmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \\ N_5 \end{pmatrix} \tag{2.7}$$

The displacements $\boldsymbol{u}$ at any points of the element are interpolated from the nodal coordinate, in the same manner that was completed for the coordinate $\boldsymbol{X}$ above. Then the displacement field can be shown as:

$$\boldsymbol{u}(\xi) = \begin{pmatrix} u_x(\xi) \\ u_y(\xi) \\ u_z(\xi) \end{pmatrix} = (\boldsymbol{u}^e)^T N = \begin{pmatrix} u_{x1} & u_{x2} & u_{x3} & u_{x4} & u_{x5} \\ u_{y1} & u_{y2} & u_{y3} & u_{y4} & u_{y5} \\ u_{z1} & u_{z2} & u_{z3} & u_{z4} & u_{z5} \end{pmatrix} \begin{pmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \\ N_5 \end{pmatrix} \tag{2.8}$$

This approach is called the isoparametric concept, which makes it possible to present the parameters at any given point within an element. The magnitude of parameters at a one of such points can be calculated through both the known values of parameters and shape functions at each node. An example of displacements can be formed like this:

$$\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} = (\boldsymbol{u}^e)^T \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{X}} = (\boldsymbol{u}^e)^T \tilde{\boldsymbol{B}}_0 \tag{2.9}$$

The deviation of a parameter can be passed to the shape function. Through this approach, we can define a new term $\tilde{\boldsymbol{B}}_0 = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}}$. The expansion of $\tilde{\boldsymbol{B}}_0$ can be derived as:

$$\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{X}} = \tilde{\boldsymbol{B}}_0 = \begin{pmatrix} \frac{\partial N_1}{\partial X} & \frac{\partial N_1}{\partial Y} & \frac{\partial N_1}{\partial Z} \\ \frac{\partial N_2}{\partial X} & \frac{\partial N_2}{\partial Y} & \frac{\partial N_2}{\partial Z} \\ \frac{\partial N_3}{\partial X} & \frac{\partial N_3}{\partial Y} & \frac{\partial N_3}{\partial Z} \\ \frac{\partial N_4}{\partial X} & \frac{\partial N_4}{\partial Y} & \frac{\partial N_4}{\partial Z} \\ \frac{\partial N_5}{\partial X} & \frac{\partial N_5}{\partial Y} & \frac{\partial N_5}{\partial Z} \end{pmatrix} = \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}} \frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{X}} \tag{2.10}$$

The first term $\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}}$ in Equation 2.10 can be directly expressed with shape function relating coordinate ξ. It is shown as:

$$\frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}} = \begin{pmatrix} \frac{\partial N_1}{\partial \xi_1} & \frac{\partial N_1}{\partial \xi_2} & \frac{\partial N_1}{\partial \xi_3} \\ \frac{\partial N_2}{\partial \xi_1} & \frac{\partial N_2}{\partial \xi_2} & \frac{\partial N_2}{\partial \xi_3} \\ \frac{\partial N_3}{\partial \xi_1} & \frac{\partial N_3}{\partial \xi_2} & \frac{\partial N_3}{\partial \xi_3} \\ \frac{\partial N_4}{\partial \xi_1} & \frac{\partial N_4}{\partial \xi_2} & \frac{\partial N_4}{\partial \xi_3} \\ \frac{\partial N_5}{\partial \xi_1} & \frac{\partial N_5}{\partial \xi_2} & \frac{\partial N_5}{\partial \xi_3} \end{pmatrix} \tag{2.11}$$

The second term $\frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{X}}$ is not straightforward to obtain. However, it is much easier to get the matrix $\frac{\partial \boldsymbol{X}}{\partial \boldsymbol{\xi}}$. This matrix is called Jacobian matrix $\boldsymbol{J}$, which is a matrix of $\boldsymbol{\xi}$ with respect to x. The inverse Jacobian matrix is denoted as $\boldsymbol{J}^{-1}$, which is the second term in the Equation 2.10.
The deformation gradient $\boldsymbol{F}$ describes the mapping of an infinitesimal fiber from its initial state to its new position in the current configuration. $\boldsymbol{F} = \boldsymbol{H} + \boldsymbol{I}$ can be derived through the transient matrix $\boldsymbol{H} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} = (\boldsymbol{u}^e)^T \tilde{\boldsymbol{B}}_0$. The Green-Lagrange strain tensor can also be written as:

$$\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{H} + \boldsymbol{H}^T + \boldsymbol{H}^T\boldsymbol{H}\right) \tag{2.12}$$

$$\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{F}^T\boldsymbol{F} - \boldsymbol{I}\right) \tag{2.13}$$

The transient matrix $\boldsymbol{H}$ can be expressed in more details as:

$$\boldsymbol{H} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{X}} = \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{\xi}} \frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{X}} = (\boldsymbol{u}^e)^T \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}} \left[\left(\boldsymbol{X}^e\right)^T \frac{\partial \boldsymbol{N}}{\partial \boldsymbol{\xi}} \right]^{-1} \tag{2.14}$$

When we have result for strain $\boldsymbol{E}$, It is possible to calculate stress $\boldsymbol{S}$. One simply constitutive equation between stress $\boldsymbol{S}$ (2. Piola-Kirchhoff-Stress tensor) and $\boldsymbol{E}$ (Green-Lagrange-Strain tensor) can be formulated as:

$$\boldsymbol{S} = \boldsymbol{C} : \boldsymbol{E} \tag{2.15}$$

The value of stress is subsequently transferred into the degree of freedom at each nodes. The principle for stress calculation can be formulated as follows:
In the Total Lagrange approach, the principle of virtual work represents that the internal strain energy $\delta \boldsymbol{W}_{int} = \int \boldsymbol{\sigma} : \delta\boldsymbol{\epsilon} \mathrm{d}\Omega_0 = \int \boldsymbol{S} : \delta\boldsymbol{E} \mathrm{d}\Omega_0$ equals the external energy $\delta \boldsymbol{W}_{ext} = (\delta \boldsymbol{u}^{e,\upsilon})^T \boldsymbol{f}^{\upsilon}_{nl}$, which is caused by external nodal force. The complete equation can be expressed as:

$$\delta \boldsymbol{W} = \delta \boldsymbol{W}_{ext} = (\delta \boldsymbol{u}^{e,\upsilon})^T \boldsymbol{f}^{\upsilon}_{int} = \int \boldsymbol{S} : \delta\boldsymbol{E} \mathrm{d}\Omega_0 = \int \left(\delta \boldsymbol{E}^{\upsilon}\right)^T \boldsymbol{S}^{\upsilon} \Omega_0 \tag{2.16}$$

The internal deformation energy can be computed by matrix-vector-product in voigt-notation or direct product of two matrix with notation (:). Now, we evaluate the variation of Green-Lagrange strain tensor $\delta\boldsymbol{E}$. From equation 2.13, it is obvious that the variation of tensor $\delta\boldsymbol{E}$ is determined by the variation of deformation gradient. According to above equation 2.14, the variation of deformation gradient can be transformed into:

$$\delta \boldsymbol{F} = \delta \boldsymbol{H} = (\delta \boldsymbol{u}^{e})^T \frac{\delta \boldsymbol{N}}{\delta \boldsymbol{X}} = (\delta \boldsymbol{u}^{e})^T \tilde{\boldsymbol{B}}_0 \tag{2.17}$$

Consequently, the variation of tensor $\delta\boldsymbol{E}$ is:

$$\delta \boldsymbol{E} = \frac{1}{2} \left(\delta \boldsymbol{F}^T \boldsymbol{F} + \boldsymbol{F}^T \delta F\right) \tag{2.18}$$

Green-Lagrange-Strain $\delta\boldsymbol{E}$ is often represented in a complex matrix form, thus it is often expressed in Voigt-Notation in computer programming. Then, $\delta\boldsymbol{E}^{\upsilon}$ can be coupled with $\boldsymbol{B}_0$-matrix as:

$$\delta \boldsymbol{E}^{\upsilon} = \boldsymbol{B}_0 \delta \boldsymbol{u}^{e,\upsilon} \tag{2.19}$$

The entries of $\boldsymbol{B}_0$ will be packed as a black box and direct implemented in the FEM simulation. Gathering the equation 2.16 and equation 2.19, the non-linear force results in:

$$\delta \boldsymbol{W} = (\delta \boldsymbol{u}^{e,\upsilon})^T \boldsymbol{f}^{\upsilon}_{int} = \int \left(\delta \boldsymbol{E}^{\upsilon}\right)^T \boldsymbol{S}^{\upsilon} \mathrm{d}\Omega_0 = (\delta \boldsymbol{u}^{e,\upsilon})^T \int \boldsymbol{B}_0^T \boldsymbol{S}^{\upsilon} \mathrm{d}\Omega_0 \tag{2.20}$$

$$\boldsymbol{f}^{\upsilon}_{int} = \int \boldsymbol{B}_0^T \boldsymbol{S}^{\upsilon} \mathrm{d}\Omega_0 \tag{2.21}$$

The non-linear internal force of element concerning to coordinate of node can be integrated by $\boldsymbol{B}_0$-martrix with second Piola-Kirchhoff-Stress tensor in voigt-notation. To obtain the tangential

stiffness matrix, it is necessary to calculate the partial derivative of internal force $\boldsymbol{f}_{int}^{v}$ respect to the nodal degree of freedom $\boldsymbol{u}^{e,v}$:

$$\frac{\partial \boldsymbol{f}_{int}^{v}}{\partial \boldsymbol{u}^{e,v}} = \boldsymbol{K} = \int \frac{\partial \boldsymbol{B}_0^T}{\partial \boldsymbol{u}^{e,v}} \boldsymbol{S}^v \mathrm{d}\Omega_0 + \int \boldsymbol{B}_0^T \frac{\partial \boldsymbol{S}^v}{\partial \boldsymbol{u}^{e,v}} \mathrm{d}\Omega_0 = \boldsymbol{K}_{geo} + \boldsymbol{K}_{mat} \tag{2.22}$$

Stiffness can be divide into two term, one for material stiffness matrix $\boldsymbol{K}_geo$, which is shown as:

$$\boldsymbol{K}_{mat} = \int \boldsymbol{B}_0^T \frac{\partial \boldsymbol{S}^v}{\partial \boldsymbol{u}^{e,v}} \mathrm{d}\Omega_0 = \int \boldsymbol{B}_0^T \frac{\partial \boldsymbol{S}^v}{\partial \boldsymbol{E}^v} \frac{\partial \boldsymbol{E}^v}{\partial \boldsymbol{u}^{e,v}} \mathrm{d}\Omega_0 = \int \boldsymbol{B}_0^T \boldsymbol{C}^{SE} \boldsymbol{B}_0 \mathrm{d}\Omega_0 \tag{2.23}$$

The derivative of stress respect to strain is considered as Tangent-Modulus $\boldsymbol{C}^{SE} = \frac{\partial \boldsymbol{S}^v}{\partial \boldsymbol{E}^v}$ by constitutive equation. The derivative of strain is $\boldsymbol{B}_0$ as determined in equation 2.13.
The geometrical stiffness matrix is much more complex to derive. In a word, the internal work from equation 2.16 is reformulated by concerning relation between the deformation gradient and continuum mechanics $\boldsymbol{P} = \boldsymbol{S}\boldsymbol{F}^T$:

$$\delta \boldsymbol{W} = \boldsymbol{u}^e : \boldsymbol{f}_{int} = \int \delta \boldsymbol{F}^T : \boldsymbol{P} \mathrm{d}\Omega_0 = \int \delta \boldsymbol{F}_{ij} \boldsymbol{P}_{ji} \mathrm{d}\Omega_0 \tag{2.24}$$

The variation of deformation gradient $\delta \boldsymbol{F} = \delta \boldsymbol{u}^{eT} \boldsymbol{B}_0$. In index-notation form it can be written as $\delta \boldsymbol{F}_{ij} = \delta \boldsymbol{u}_{ki}^e \tilde{\boldsymbol{B}_{kj}^0}$. Then, the internal force in matrix notation can be derived as:

$$\delta \boldsymbol{W} = \delta u^e : \boldsymbol{f}_{int} = \boldsymbol{u}_{ki}^e \boldsymbol{f}_{ki}^{int} = \int \delta \boldsymbol{F}_{ji} \boldsymbol{P}_{ij} \mathrm{d}\Omega_0 = \delta \boldsymbol{u}_{ki}^e \int \tilde{\boldsymbol{B}_{kj}^0} \boldsymbol{P}_{ji} \mathrm{d}\Omega_0 = \delta \boldsymbol{u}^e : \int \tilde{\boldsymbol{B}}_0 \boldsymbol{P} \mathrm{d}\Omega_0 \tag{2.25}$$

$$\boldsymbol{f}_{int} = \int \tilde{\boldsymbol{B}}_0 \boldsymbol{P} \mathrm{d}\Omega_0 \tag{2.26}$$

The tangential stiffness matrix is defined by changing the force vector, which is determined over time concerning with the velocity of nodal displacement $\dot{\boldsymbol{u}^e}$. Because Jacobi-Matrix is relative complex to build. Therefore, the derivative of internal force $\dot{\boldsymbol{f}_{int}}$ in time as follows:

$$\dot{\boldsymbol{f}_{int}} = \int \tilde{\boldsymbol{B}}_0 \dot{\boldsymbol{S}} \boldsymbol{F}^T \mathrm{d}\Omega_0 + \int \tilde{\boldsymbol{B}}_0 \boldsymbol{S} \dot{\boldsymbol{F}^T} \mathrm{d}\Omega_0 \tag{2.27}$$

The fist term conresponds to material stiffness, becasue the time derivative of second Piola-Kirchhoff-Stress tensor is related with time-dependent part. And the second term conresponds to geometrical stiffness as shown in equation 2.22. Becase the time derivative $\dot{\boldsymbol{F}}$ equals $\dot{\boldsymbol{u}^e}^T \tilde{\boldsymbol{B}}_0$, equation2.27 can be transformed as:

$$\dot{\boldsymbol{f}}_{int,geo} = \int \tilde{\boldsymbol{B}}_0 \boldsymbol{S} \tilde{\boldsymbol{B}}_0 \Omega_0 \dot{\boldsymbol{u}}^e \tag{2.28}$$

The geometrical stiffness matrix is given, because the temporal variation of non-linear force is coupled with the displacements over the tangential stiffness matrix.

$$\dot{\boldsymbol{f}}_{int,geo} = \boldsymbol{K}_{geo} \dot{\boldsymbol{u}}^e = \int \tilde{\boldsymbol{B}}_0 \boldsymbol{S} \tilde{\boldsymbol{B}}_0 \Omega_0 \dot{\boldsymbol{u}}^e \tag{2.29}$$

The last step to determine the non-linear force and the tangential stiffness matrix is to integrate on domain $d\Omega_0$. There are two concept for this integration to mention: the first one is transformation from domain $d\Omega_0$ to reference coordinate system:

$$\int f(x)\,\mathrm{d}\Omega = \int f(x)\,\frac{\partial \xi}{\partial \xi}\mathrm{d}\Omega = \int f(x)\,\frac{\partial \Omega}{\partial \xi}\mathrm{d}\xi = \int f(x)\,det\left(\frac{\partial X}{\partial \xi}\right)\mathrm{d}\xi \tag{2.30}$$

For the transformation of integration, it is necessary to adapt the bound of integration. It is in FEM a pleasant benefit, its advantage is the new bound of integration is the standard bound of reference element. So it is easier to determine the bound of integration. And the Jocobi-matrix is ready to use, because we have once the inverse of Jocobi-matrix calculated.

2.3.1 Gauss Integration

Numerical integration plays a important role in finite element. Gauss integration, as an efficient approach compared to other numerical integration, integrate a function $\boldsymbol{f}(\boldsymbol{\xi})$ on the spatial domain Ω is replaced by a summation of certain function values at the so-called Gauss points $\tilde{\boldsymbol{\xi}}$ which are each multiplied by a scalar (weight) $\boldsymbol{\omega}$. In total $numgp_1 \times numggp_2 \times numgp_3$ Gauss points are used:

$$\int_E f(\xi)\,\mathrm{d}\xi_1\mathrm{d}\xi_2\mathrm{d}\xi_3 = \sum_{i=1}^{numgp_1}\sum_{j=1}^{numgp_2}\sum_{k=1}^{numgp_3} f\left(\tilde{\xi}_1^i\tilde{\xi}_2^j\tilde{\xi}_3^k\right)\cdot w_1^i w_2^j w_3^k \tag{2.31}$$

In which we used corresponding weights for each of the three spatial directions. This allows to use Gauss points coordinates and weights tabulated for the one-dimensional case. The location and weights of these points can be given analytically in Table 2.1.

2.4 Extrapolation from Gauss Points

2.4.1 Quad4

Quad4 is used for 2-D modelling of solid structure. The element can be used either as a plane element (plane stress or plane strain) or as an axisymmetric element. The element is defined by four nodes having two degrees of freedom at each node: translations in the nodal x and y directions. The element has plasticity, creep, swelling, large deflection, and large strain capabilities. The normal Gauss integration rule for element stiffness evaluation is 2×2, as illustrated in Figure 2.3. The components(strains, stresses) are calculated at the Gauss points, which are identified as k_1^G, k_2^G, k_3^G and k_4^G in Figure 2.3. Point k_i^G is closest to node k_i^E so it is seen that Gauss point numbering essentially follows element node numbering in the counterclockwise sense. The natural coordinates of these points are listed in Table 2.2. The components are calculated at these Gauss points by passing these natural coordinates to the shape function subroutine. Then each strain and stress component is transported to the corner nodes k_1^E through k_4^E by a bilinear extrapolation based on the computed values at k_1^G through k_4^E. To understand the extrapolation procedure more clearly it is convenient to consider the region bounded by the Gauss points as an

Table 2.1: Gauss-Legendre points and weights

numgp	i	$\tilde{\xi}^i$	ω^i
1	1	0	2
2	1	$-1/\sqrt{3}$	1
	2	$+1/\sqrt{3}$	1
3	1	$-\sqrt{3/5}$	5/9
	2	0	8/9
	3	$+\sqrt{3/5}$	5/9
4	1	$-\sqrt{\left(15+\sqrt{120}\right)/35}$	$\left(18-\sqrt{30}\right)/36$
	2	$-\sqrt{\left(15-\sqrt{120}\right)/35}$	$\left(18+\sqrt{30}\right)/36$
	3	$+\sqrt{\left(15-\sqrt{120}\right)/35}$	$\left(18+\sqrt{30}\right)/36$
	4	$+\sqrt{\left(15+\sqrt{120}\right)/35}$	$\left(18-\sqrt{30}\right)/36$
5	1	$-1/3\sqrt{5+2\sqrt{10/7}}$	$\left(332-13\sqrt{70}\right)/900$
	2	$-1/3\sqrt{5-2\sqrt{10/7}}$	$\left(332+13\sqrt{70}\right)/900$
	3	0	128/255
	4	$+1/3\sqrt{5-2\sqrt{10/7}}$	$\left(332+13\sqrt{70}\right)/900$
	5	$+1/3\sqrt{5+2\sqrt{10/7}}$	$\left(332-13\sqrt{70}\right)/900$

internal Gauss element. This interpretation is depicted in Figure 2.3. The element is denoted by (E). The internal Gauss element, denoted by (G), is also a four-node quadrilateral. The coordinate of node for element and Gauss element can be represented as k_i^G and k_i^E, respectively. Its quadrilateral (natural) coordinates are denoted by ξ' and η'. These are linked to ξ' and η' by the simple relations from Gauss-Legendre quadrature in Table 2.2.

$$k_i^G = \frac{k_i^E}{\sqrt{3}}, \quad k_i^E = k_i^G \sqrt{3} \tag{2.32}$$

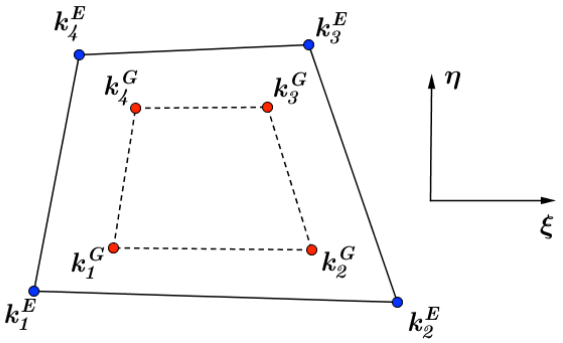


Figure 2.3: Quad4 in element coordinate and Gauss element coordinate.

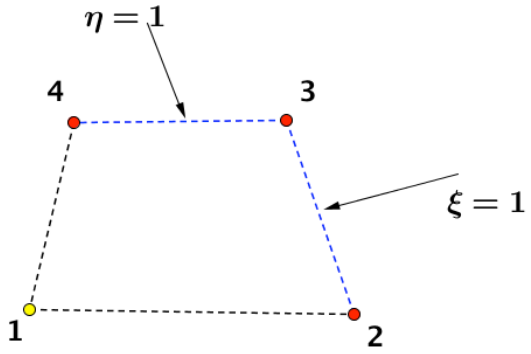


Figure 2.4: Equation of side opposite corner 1 for Quad4.

The element geometry and natural coordinates are shown in Figure 2.5. There is only one type of node and associated shape function. We consider node 1 as typical node. From Figure 2.4 we can suggest:

$$N_1^e = c_1 L_{2-3} L_{3-4} \tag{2.33}$$

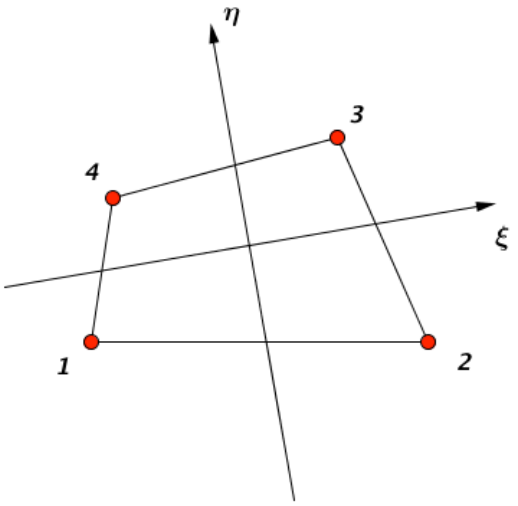


Figure 2.5: Equation of side opposite corner 1 for Quad4.

The equation of side 2-3 is $\xi - 1 = 0$. The equation of side 3-4 is $\eta - 1 = 0$. Replacing in Equation (2.33):

$$N_1^e(\xi,\eta) = c_1(\xi - 1)(\eta - 1) = c_1(1-\xi)(1-\eta) \tag{2.34}$$

We evaluate the point at node 1 to find c_1, the natural coordinates can be expressed as $\xi = \eta = -1$:

$$N_1^e(-1,-1) = c_1 \times 2 \times 2 = 4c_1 = 1 \tag{2.35}$$

So, $c_1 = \frac{1}{4}$ and the shape functions is

$$N_1^e = \frac{1}{4}(1-\xi)(1-\eta) \tag{2.36}$$

We can use the same approach to calculate the other three nodes. Following this general expression, the shape functions of Node 2, 3 and 4 are demonstrated as

$$N_2^e = \frac{1}{4}(1+\xi)(1-\eta) \tag{2.37}$$

$$N_3^e = \frac{1}{4}(1+\xi)(1+\eta) \tag{2.38}$$

$$N_4^e = \frac{1}{4}(1-\xi)(1+\eta) \tag{2.39}$$

When we have all the shape function for Gauss element, it is able to extrapolate the component(stress, strain, etc) from Gauss points k_i^G to corner nodes k_i^E. According to 2.2 and 2.3,

Table 2.2: Natural Coordinate of Quad4

Corner node	ξ	η	ξ'	η'	Gauss node	ξ	η	ξ'	η'
1	-1	-1	$-\sqrt{3}$	$-\sqrt{3}$	1'	$+1/\sqrt{3}$	$-1/\sqrt{3}$	-1	-1
2	+1	-1	$+\sqrt{3}$	$-\sqrt{3}$	2'	$+1/\sqrt{3}$	$+1/\sqrt{3}$	+1	-1
3	+1	+1	$+\sqrt{3}$	$+\sqrt{3}$	3'	$+1/\sqrt{3}$	$+1/\sqrt{3}$	+1	+1
4	-1	+1	$-\sqrt{3}$	$+\sqrt{3}$	4'	$-1/\sqrt{3}$	$+1/\sqrt{3}$	-1	+1

we have the corner nodes in Gauss coordinate: $k_1^E(-\sqrt{3},-\sqrt{3})$, $k_2^E(\sqrt{3},-\sqrt{3})$, $k_3^E(\sqrt{3},\sqrt{3})$, $k_4^E(\sqrt{3},\sqrt{3})$. The extrapolation process is to replace all the corner nodes in Gauss coordinate into the shape function of each Gauss points. Then we obtain:

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix} = \begin{pmatrix} 1+\frac{1}{2}\sqrt{3} & -\frac{1}{2} & 1-\frac{1}{2}\sqrt{3} & -\frac{1}{2} \\ -\frac{1}{2} & 1+\frac{1}{2}\sqrt{3} & -\frac{1}{2} & 1-\frac{1}{2}\sqrt{3} \\ 1-\frac{1}{2}\sqrt{3} & -\frac{1}{2} & 1+\frac{1}{2}\sqrt{3} & -\frac{1}{2} \\ -\frac{1}{2} & 1-\frac{1}{2}\sqrt{3} & -\frac{1}{2} & 1+\frac{1}{2}\sqrt{3} \end{pmatrix} \begin{pmatrix} w_1{}' \\ w_2{}' \\ w_3{}' \\ w_4{}' \end{pmatrix} \tag{2.40}$$

Here w' means strains and stresses for our case. So the w' part can be written as:

$$\begin{pmatrix} w_1{}' \\ w_2{}' \\ w_3{}' \\ w_4{}' \end{pmatrix} = \begin{pmatrix} \epsilon_{1xx} & \epsilon_{1yy} & \epsilon_{1zz} & 2\epsilon_{1yz} & 2\epsilon_{1xz} & 2\epsilon_{1xy} \\ \epsilon_{2xx} & \epsilon_{2yy} & \epsilon_{2zz} & 2\epsilon_{2yz} & 2\epsilon_{2xz} & 2\epsilon_{2xy} \\ \epsilon_{3xx} & \epsilon_{3yy} & \epsilon_{3zz} & 2\epsilon_{3yz} & 2\epsilon_{3xz} & 2\epsilon_{3xy} \\ \epsilon_{4xx} & \epsilon_{4yy} & \epsilon_{4zz} & 2\epsilon_{4yz} & 2\epsilon_{4xz} & 2\epsilon_{4xy} \end{pmatrix} \tag{2.41}$$

$$\begin{pmatrix} w_1' \\ w_2' \\ w_3' \\ w_4' \end{pmatrix} = \begin{pmatrix} \sigma_{1xx} & \sigma_{1yy} & \sigma_{1zz} & \sigma_{1yz} & \sigma_{1xz} & \sigma_{1xy} \\ \sigma_{2xx} & \sigma_{2yy} & \sigma_{2zz} & \sigma_{2yz} & \sigma_{2xz} & \sigma_{2xy} \\ \sigma_{3xx} & \sigma_{3yy} & \sigma_{3zz} & \sigma_{3yz} & \sigma_{3xz} & \sigma_{3xy} \\ \sigma_{4xx} & \sigma_{4yy} & \sigma_{4zz} & \sigma_{4yz} & \sigma_{4xz} & \sigma_{4xy} \end{pmatrix} \tag{2.42}$$

2.4.2 Quadrangle with quadratic shape function and eight nodes: Quad8

Quad8 is a higher order 2-D, 8 node element. This element is defined by 8 nodes having two degrees of freedom at each node: translation in the nodal x and y direction. The element may be used as a plane element (plane stress, plane strain) or as an axisymmetric element. For Quad8 we have various choice for the type of Gauss element. We usually use four, eight and nine nodes Gauss element for three quadrilateral elements. Here we introduce the Gauss element with nine nodes. The nine-nodes quadrilateral has three types of shape functions, which are associated with corner, midpoint nodes and center node, respectively. The element coordinate and Gauss element coordinate are illustrated in Figure 2.6
The three types of shape functions are illustrated in Figure for nodes 1, 5 and 9, respectively. The procedure for calculating shape function has been clearly expressed in Section 2.4.1. Here the summary of calculation for nodes 1, 5 and 9, which are taken as three typical types: The three types of shape function are illustrated in Figure 2.7.

$$N_1^e = c_1 L_{2-3} L_{3-4} L_{5-7} L_{6-8} = c_1 (\xi - 1)(\eta - 1)\xi\eta \tag{2.43}$$

$$N_5^e = c_5 L_{2-3} L_{1-4} L_{6-8} L_{3-4} = c_5 (\xi - 1)(\xi + 1)\eta(\eta - 1) = c_5 \left(1 - \xi^2\right)\eta(1 - \eta) \tag{2.44}$$

$$N_9^e = c_9 L_{1-2} L_{2-3} L_{3-4} L_{4-1} = c_9 (\xi - 1)(\eta - 1)(\xi + 1)(\eta + 1) = c_9 \left(1 - \xi^2\right)\left(1 - \eta^2\right) \tag{2.45}$$

Applying the normalization conditions results in:

$$c_1 = \frac{1}{4}, \quad c_5 = -\frac{1}{2}, c_9 = 1$$

By following this approach all the shape functions can be calculated:

$$N_1 = \frac{1}{4}(\xi - 1)(\eta - 1) \cdot \xi \cdot \eta \tag{2.46}$$

$$N_2 = \frac{1}{4}(\xi + 1)(\eta - 1) \cdot \xi \cdot \eta \tag{2.47}$$

$$N_3 = \frac{1}{4}(\xi + 1)(\eta + 1) \cdot \xi \cdot \eta \tag{2.48}$$

$$N_4 = \frac{1}{4}(\xi - 1)(\eta + 1) \cdot \xi \cdot \eta \tag{2.49}$$

$$N_5 = \frac{1}{2}(1+\xi)(1+\xi)(\eta-1)\cdot\eta \tag{2.50}$$

$$N_6 = \frac{1}{2}(1+\eta)(1-\eta)(\xi+1)\cdot\xi \tag{2.51}$$

$$N_7 = \frac{1}{2}(1+\xi)(1+\xi)(\eta+1)\cdot\eta \tag{2.52}$$

$$N_8 = \frac{1}{2}(1+\eta)(1-\eta)(\xi-1)\cdot\xi \tag{2.53}$$

$$N_9 = (1+\xi)(1-\xi)(1+\eta)(1-\eta) \tag{2.54}$$

Same as Quad4, the extrapolation function can be expressed by replacing corner nodes in Gauss coordinate into the shape function of Gauss points:

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_3 & a_2 & a_4 & a_5 & a_5 & a_4 & a_6 \\ a_2 & a_1 & a_2 & a_3 & a_4 & a_4 & a_5 & a_5 & a_6 \\ a_3 & a_2 & a_1 & a_2 & a_5 & a_4 & a_4 & a_5 & a_6 \\ a_2 & a_3 & a_2 & a_1 & a_5 & a_5 & a_4 & a_4 & a_6 \\ 0 & 0 & 0 & 0 & a_7 & 0 & a_8 & 0 & a_9 \\ 0 & 0 & 0 & 0 & 0 & a_7 & 0 & a_8 & a_9 \\ 0 & 0 & 0 & 0 & a_8 & 0 & a_7 & 0 & a_9 \\ 0 & 0 & 0 & 0 & 0 & a_8 & 0 & a_7 & a_9 \end{pmatrix} \begin{pmatrix} w_1{}' \\ w_2{}' \\ w_3{}' \\ w_4{}' \\ w_5{}' \\ w_6{}' \\ w_7{}' \\ w_8{}' \\ w_9{}' \end{pmatrix} \tag{2.55}$$

$a_1 = +2.1869398$
$a_2 = +0.2777778$
$a_3 = +0.0352824$
$a_4 = -0.9858870$
$a_5 = -0.1252241$
$a_6 = +0.4444444$
$a_7 = +1.4788331$
$a_8 = +0.1878361$
$a_9 = -0.6666666$

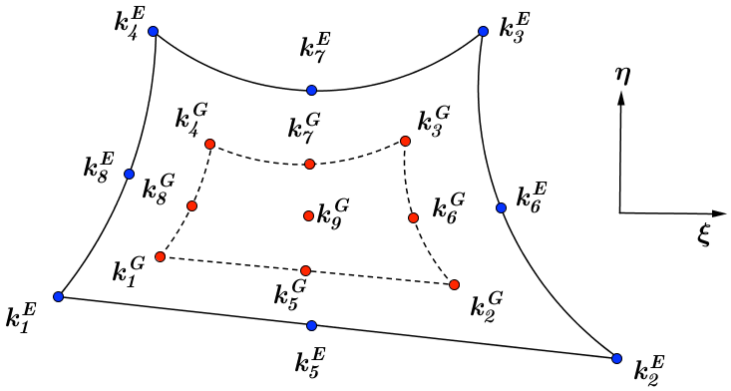


Figure 2.6: Quad8 in element coordinate and Gauss element coordinate.

2.4.3 Triangle with three nodes: Tri3

Tri3 is used for 2-D modelling of solid structure. This element is defined by 3 nodes having two degrees of freedom at each node: translation in the nodal x and y direction. The geometry of the 3-node triangle shown in Figure 2.8 is specified by the location of its three corner nodes on the $\{x, y\}$ plane. The shape function for triangular element has a different form, which compares with quadrilateral elements. The three shape functions have the simply own coordinates - the triangular coordinates: $N_i = \xi_i$ for $i = 1, 2, 3$. The shape function can be derived from a method as follows: The equation of triangle side opposite to node i is $L_{j-k} = \xi_i = 0$, where j and k are the cyclic permutations of i. Here symbol L_{j-k} denotes the left hand side of the homogeneous equation of the natural coordinate line that passes through node points j and k. See Figure 2.9 for $i = 1$, $j = 2$ and $k = 3$. Hence the obvious suppose is:

$$N_i^e = c_i L_i \tag{2.56}$$

To find c_1, evaluate $N_1^e\left(\xi_1, \xi_2, \xi_3\right)$ at node 1. The triangular coordinates of this node are ξ_1 =1, $\xi_2 = \xi_3 = 0$. $N_1^e(1, 0, 0) = c_1 \times 1 = 1$. so $c_1 = 1$ and analogically $c_2 = 1$, $c_3 = 1$

$$N_1^e = L_1, \quad N_2^e = L_2, \quad N_3^e = L_3 \tag{2.57}$$

Combining the shape function and corner coordinate in Gauss element from Table 2.3, the extrapolation of Tri3 can be written as:

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} 5/3 & -1/3 & -1/3 \\ -1/3 & 5/3 & -1/3 \\ -1/3 & -1/3 & 5/3 \end{pmatrix} \begin{pmatrix} w_1{}' \\ w_2{}' \\ w_3{}' \end{pmatrix} \tag{2.58}$$

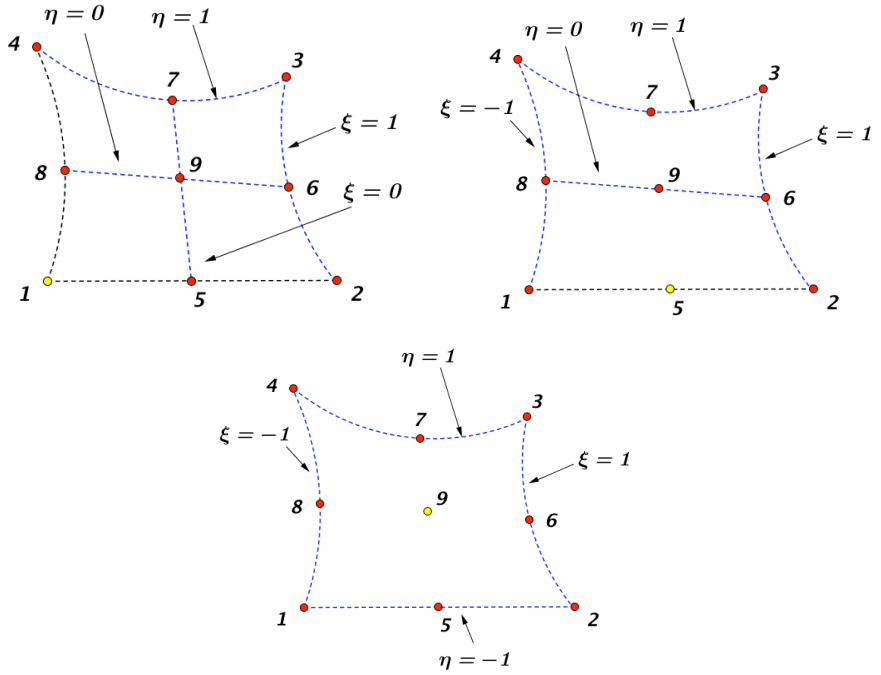


Figure 2.7: Equation of side opposite corner 1 for Quad8.

Quantities that are closely related with the element geometry are best represented in triangular coordinates. On the other hand, quantities such as displacements, strains and stresses are often calculated in the Cartesian system. Thus we need transformation equations through which it is possible to pass from one coordinate system to the other. Cartesian and triangular coordinates are linked by the relation.

$$\begin{pmatrix} 1 \\ x \\ y \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_2 \\ y_1 & y_2 & y_2 \end{pmatrix} \begin{pmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{pmatrix} \tag{2.59}$$

$$\begin{pmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{pmatrix} = \frac{1}{2A} \begin{pmatrix} x_2y_3 - x_3y_2 & y_2 - y_3 & x_3 - x_2 \\ x_3y_1 - x_1y_3 & y_3 - y_1 & x_1 - x_3 \\ x_1y_2 - x_2y_1 & y_1 - y_2 & x_2 - x_1 \end{pmatrix} \begin{pmatrix} 1 \\ x \\ y \end{pmatrix} = \frac{1}{2A} \begin{pmatrix} 2A_{23} & y_{23} & x_{32} \\ 2A_{31} & y_{31} & x_{13} \\ 2A_{12} & y_{12} & x_{21} \end{pmatrix} \begin{pmatrix} 1 \\ x \\ y \end{pmatrix} \tag{2.60}$$

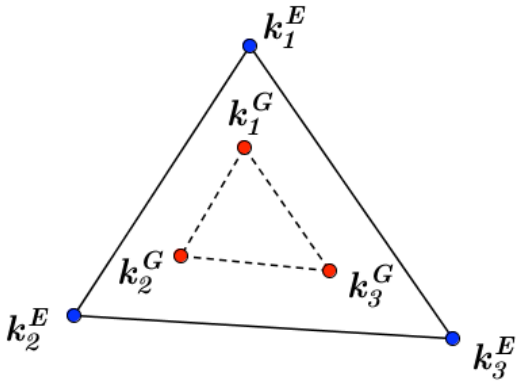


Figure 2.8: Tri3 in element coordinate and Gauss element coordinate.

2.4.4 Second order triangle with six nodes: Tri6

Tri6 is a higher order 2-D, 8 node element. This element is defined by 6 nodes having two degrees of freedom at each node: translation in the nodal x and y direction. The geometry of six-node quadratic is shown in Figure 2.10. Two kinds of Gauss elements can be applied in this case: triangle with three nodes and triangle with six nodes. Here we explain the case with six nodes. Inspection reveals two types of nodes: corners(1, 2 and 3) and midside nodes(4, 5 and 6). For both cases it is necessary to product the two linear functions in the triangular coordinates because the shape function should be quadratic. These functions are illustrated in Figures 2.11 for corner node 1 and midside node 4, respectively.
For corner node 1, inspection of Figure 2.11 at top left side suggests trying

$$N_1^e = c_1 L_{2-3} L_{4-6} \tag{2.61}$$

N_1^e will vanish over 2-5-3 and 4-6. This makes the function zero at node 2,3,4,5,6, as is obvious upon inspection of Figure 2.11, while being nonzero at 1. The value can be adjusted to be unity if c_1 is appropriately chosen. The equations of the lines that appear in Equation 2.61 are

$$L_{2-3} : \xi_1 = 0, \quad L_{4-6} : \xi_1 - \frac{1}{2} = 0 \tag{2.62}$$

Replacing into Equation 2.61

$$N_1^e = c_1 \xi_1 \left(\xi_1 - \frac{1}{2}\right) \tag{2.63}$$

Same as triangle with three nodes, $N_1^e(1,0,0) = c_1 \times 1 \times \frac{1}{2} = 1$. Then $c_1 = 2$ can be calculated and finally

$$N_1^e = 2\xi_1 \left(\xi_1 - \frac{1}{2}\right) = \xi_1 \left(2\xi_1 - 1\right) \tag{2.64}$$

Table 2.3: Natural Coordinate of Tri3

Corner node	ξ	η	ξ'	η'	Gauss node	ξ	η	ξ'	η'
1	-1	-1	-5/3	-5/3	1'	-2/3	-2/3	-1	-1
2	+1	-1	+7/3	-5/3	2'	+1/3	-2/3	+1	-1
3	-1	+1	-5/3	+7/3	3'	-2/3	+1/3	-1	+1

For midside node 4, inspection of Figure 2.11 suggests trying

$$N_4^e = c_4 L_{2-3} L_{1-3} \tag{2.65}$$

The equation of sides L_{2-3} and L_{1-3} are $\xi_1 = 0$ and $\xi_2 = 0$, respectively. Therefore $N_4^e\left(\xi_1, \xi_2, \xi_3\right) = c_4 \xi_1 \xi_2$. To find c_4, evaluate this function at node 4, the triangular coordinates of which are $\xi_1 = \xi_2 = \frac{1}{2}$,$\xi_3 = 0$. Then $N_4^e\left(\frac{1}{2}, \frac{1}{2}, 0\right) = c_4 \times \frac{1}{2} \times \frac{1}{2} = 1$. Hence $c_4 = 4$, which the shape function gives

$$N_4^e = 4\xi_1 \xi_2 \tag{2.66}$$

The rest shape function can be calculated by following same approach. The rest shape functions of Tri6 can be expressed as:

$$N_2^e = \xi_2\left(2\xi_2 - 1\right) \tag{2.67}$$

$$N_3^e = \xi_3\left(2\xi_3 - 1\right) \tag{2.68}$$

$$N_5^e = 4\xi_2 \xi_3 \tag{2.69}$$

$$N_6^e = 4\xi_1 \xi_3 \tag{2.70}$$

Then the extrapolation function of Tri6 can be written as:

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{pmatrix} = \begin{pmatrix} a_1 & a_1 & a_2 & a_3 & a_4 & a_4 \\ a_1 & a_2 & a_1 & a_4 & a_4 & a_3 \\ a_2 & a_1 & a_1 & a_4 & a_3 & a_4 \\ a_1 & a_7 & a_7 & a_5 & a_6 & a_5 \\ a_7 & a_7 & a_1 & a_6 & a_5 & a_5 \\ a_7 & a_1 & a_7 & a_5 & a_5 & a_6 \end{pmatrix} \begin{pmatrix} w_1' \\ w_2' \\ w_3' \\ w_4' \\ w_5' \\ w_6' \end{pmatrix} \tag{2.71}$$

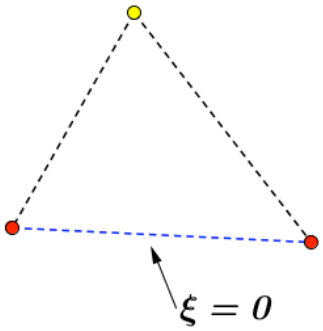


Figure 2.9: Equation of side opposite corner 1 for Tri3.

$$a_1 = +0.55555556$$
$$a_2 = +3.88888889$$
$$a_3 = +0.44444444$$
$$a_4 = -2.22222222$$
$$a_5 = -0.88888889$$
$$a_6 = +1.77777778$$
$$a_7 = +0.22222222$$

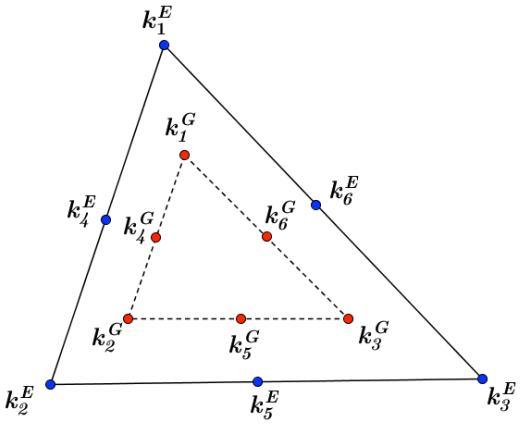


Figure 2.10: Tri6 in element coordinate and Gauss element coordinate.

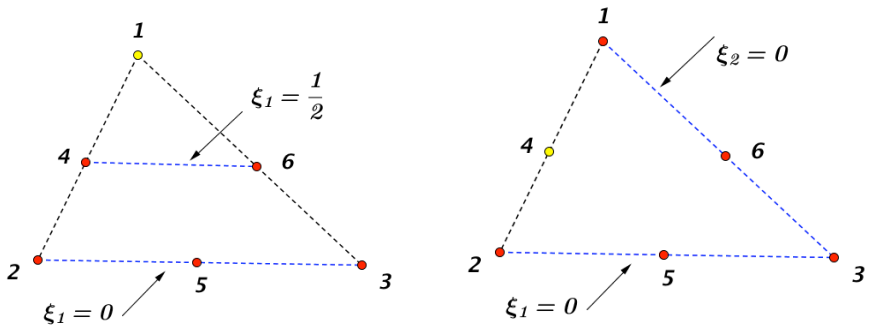


Figure 2.11: Equation of side opposite corner 1 for Tri6.

2.4.5 Tetrahedron element with four nodes: Tet4

Tet4 is a higher order 3-D 4-node solid element. The element is defined by 4 nodes having three degrees of freedom per node: translation in the nodal x, y and z directions. The geometry of Tet4 are shown in Figure 2.12. For tetrahedron element it is beneficial to use its own coordinate. The shape function of Tet4 are:

$$N_1 = L_1 \tag{2.72}$$

$$N_2 = L_2 \tag{2.73}$$

$$N_3 = L_3 \tag{2.74}$$

$$N_4 = L_4 \tag{2.75}$$

We take a tetrahedron with four nodes as Gauss element for Gauss integration. The relation between Gauss and nodal coordinate are represented in Table 2.4. From this table, we can find out four corner nodes in Gauss coordinate: $k_1^E(j,k,k,k)$, $k_2^E(k,j,k,k)$, $k_3^E(k,k,j,k)$, $k_4^E(k,k,k,j)$ ($j = 1.927051, k = -0.309017$). After replacing the corner points into shape function, the extrapolation of Tet4 are shown as:

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_2 & a_2 \\ a_2 & a_1 & a_2 & a_2 \\ a_2 & a_2 & a_1 & a_2 \\ a_2 & a_2 & a_2 & a_1 \end{pmatrix} \begin{pmatrix} w_1{}' \\ w_2{}' \\ w_3{}' \\ w_4{}' \end{pmatrix} \tag{2.76}$$

$a_1 = +1.927051$

$a_2 = -0.309017$

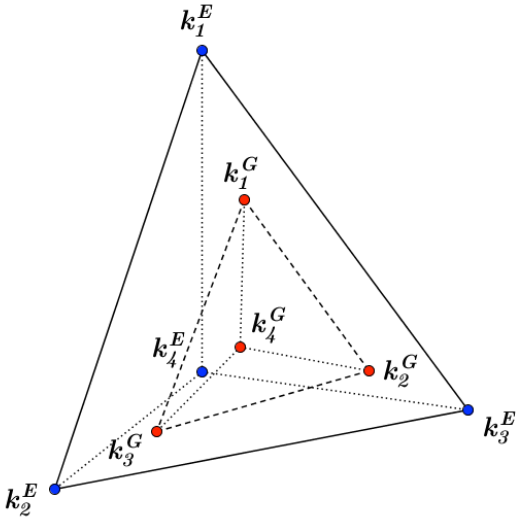


Figure 2.12: Tet4 in element coordinate and Gauss element coordinate.

2.4.6 Tetrahedron element with ten nodes: Tet10

Tet10 is a higher order 3-D 10-node solid element. The element is defined by 10 nodes having three degrees of freedom per node: translation in the nodal x, y and z directions. The geometry, node location and coordinate system for this element are shown in Figure 2.13. We choose a tetrahedron with four nodes as internal Gauss element. The shape function of Tet10 are same as Tet4:

$$N_1 = L_1 \tag{2.77}$$

$$N_2 = L_2 \tag{2.78}$$

$$N_3 = L_3 \tag{2.79}$$

$$N_4 = L_4 \tag{2.80}$$

The relation between Gauss and nodal coordinate are basically also similar as Tet4. The new appended six corner nodes are picked as midpoint of each edges. Table 2.5 shows the details about Gauss and nodal coordinate system, respectively. Then the extrapolation of Tet10 can be

Table 2.4: Tetrahedral Coordinate of Tet4

Corner node	L1	L2	L3	L4	L1'	L2'	L3'	L4'
1	1	0	0	0	α	β	β	β
2	0	1	0	0	β	α	β	β
3	0	0	1	0	β	β	α	β
4	0	0	0	1	β	β	β	α
Gauss node	L1	L2	L3	L4	L1'	L2'	L3'	L4'
1'	α	β	β	β	1	0	0	0
2'	β	α	β	β	0	1	0	0
3'	β	β	α	β	0	0	0	1
4'	β	β	β	α	0	0	0	1

$\alpha = 0.58541020$; $\beta = 0.13819660$

formed as:

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \\ w_9 \\ w_{10} \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_2 & a_2 \\ a_2 & a_1 & a_2 & a_2 \\ a_2 & a_2 & a_1 & a_2 \\ a_2 & a_2 & a_2 & a_1 \\ a_3 & a_3 & a_2 & a_2 \\ a_3 & a_2 & a_3 & a_2 \\ a_3 & a_2 & a_2 & a_3 \\ a_2 & a_3 & a_3 & a_2 \\ a_2 & a_3 & a_2 & a_3 \\ a_2 & a_2 & a_3 & a_3 \end{pmatrix} \begin{pmatrix} w_1{}' \\ w_2{}' \\ w_3{}' \\ w_4{}' \end{pmatrix} \tag{2.81}$$

$a_1 = +1.927051$
$a_2 = -0.309017$
$a_3 = +0.809017$

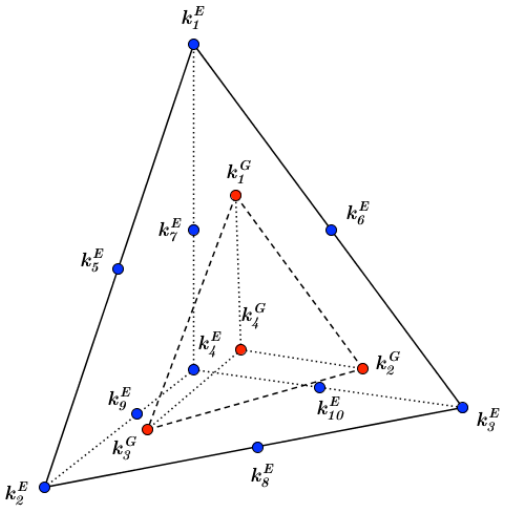


Figure 2.13: Tet10 in element coordinate and Gauss element coordinate.

2.5 Assembly

In previous section, we have solved stain and stress in one element. The next step is to assembly strain and stress at each node. At the same time, element stiffness matrix and force vector are also assembled at each degree of freedom. In FEM, the whole primary field can be seen as a continuous field, which two neighboring element share one or more nodes. Now we take a simple 2-D an object as example, which are meshed by two Tri3 element as depicted in Figure 2.14. We assume that the stiffness matrix k and force vector f for each single element have been

Table 2.5: Coordinate system of Tet10

Corner node	L1	L2	L3	L4	L1'	L2'	L3'	L4'
1	1	0	0	0	α	β	β	β
2	0	1	0	0	β	α	β	β
3	0	0	1	0	β	β	α	β
4	0	0	0	1	β	β	β	α
5	0.5	0.5	0	0	$\frac{(\alpha+\beta)}{2}$	$\frac{(\alpha+\beta)}{2}$	β	β
6	0.5	0	0.5	0	$\frac{(\alpha+\beta)}{2}$	β	$\frac{(\alpha+\beta)}{2}$	β
7	0.5	0	1	0.5	$\frac{(\alpha+\beta)}{2}$	β	β	$\frac{(\alpha+\beta)}{2}$
8	0	0.5	0.5	0	β	$\frac{(\alpha+\beta)}{2}$	$\frac{(\alpha+\beta)}{2}$	β
9	0	0.5	0	0.5	β	$\frac{(\alpha+\beta)}{2}$	β	$\frac{(\alpha+\beta)}{2}$
10	0	0	0.5	0.5	β	β	$\frac{(\alpha+\beta)}{2}$	$\frac{(\alpha+\beta)}{2}$
Gauss node	L1	L2	L3	L4	L1'	L2'	L3'	L4'
1'	α	β	β	β	1	0	0	0
2'	β	α	β	β	0	1	0	0
3'	β	β	α	β	0	0	0	1
4'	β	β	β	α	0	0	0	1

$\alpha = 0.58541020$; $\beta = 0.13819660$

done before. The stiffness matrix of elements is given as:

$$k^{(1)} = \begin{pmatrix} k_{11}^{(1)} & k_{12}^{(1)} & k_{13}^{(1)} & k_{14}^{(1)} & k_{15}^{(1)} & k_{16}^{(1)} \\ k_{21}^{(1)} & k_{22}^{(1)} & k_{23}^{(1)} & k_{24}^{(1)} & k_{25}^{(1)} & k_{26}^{(1)} \\ k_{31}^{(1)} & k_{32}^{(1)} & k_{33}^{(1)} & k_{34}^{(1)} & k_{35}^{(1)} & k_{36}^{(1)} \\ k_{41}^{(1)} & k_{42}^{(1)} & k_{43}^{(1)} & k_{44}^{(1)} & k_{45}^{(1)} & k_{46}^{(1)} \\ k_{51}^{(1)} & k_{52}^{(1)} & k_{53}^{(1)} & k_{54}^{(1)} & k_{55}^{(1)} & k_{56}^{(1)} \\ k_{61}^{(1)} & k_{62}^{(1)} & k_{63}^{(1)} & k_{64}^{(1)} & k_{65}^{(1)} & k_{66}^{(1)} \end{pmatrix} \tag{2.82}$$

$$k^{(2)} = \begin{pmatrix} k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} & k_{15}^{(2)} & k_{16}^{(2)} \\ k_{21}^{(2)} & k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} & k_{25}^{(2)} & k_{26}^{(2)} \\ k_{31}^{(2)} & k_{32}^{(2)} & k_{33}^{(2)} & k_{34}^{(2)} & k_{35}^{(2)} & k_{36}^{(2)} \\ k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{44}^{(2)} & k_{45}^{(2)} & k_{46}^{(2)} \\ k_{51}^{(2)} & k_{52}^{(2)} & k_{53}^{(2)} & k_{54}^{(2)} & k_{55}^{(2)} & k_{56}^{(2)} \\ k_{61}^{(2)} & k_{62}^{(2)} & k_{63}^{(2)} & k_{64}^{(2)} & k_{65}^{(2)} & k_{66}^{(2)} \end{pmatrix} \tag{2.83}$$

Now we can combine the stiffness matrix contribution at the same degree of freedom. The

procedure of assembly is represented as follows:

$$K = \begin{pmatrix} k_{11}^{(1)} & k_{12}^{(1)} & k_{15}^{(1)} & k_{16}^{(1)} & k_{13}^{(1)} & k_{14}^{(1)} & 0 & 0 \\ k_{21}^{(1)} & k_{22}^{(1)} & k_{25}^{(1)} & k_{26}^{(1)} & k_{23}^{(1)} & k_{24}^{(1)} & 0 & 0 \\ k_{51}^{(1)} & k_{52}^{(1)} & k_{55}^{(1)} & k_{56}^{(1)} & k_{53}^{(1)} & k_{54}^{(1)} & 0 & 0 \\ k_{61}^{(1)} & k_{62}^{(1)} & k_{65}^{(1)} & k_{66}^{(1)} & k_{63}^{(1)} & k_{64}^{(1)} & 0 & 0 \\ k_{31}^{(1)} & k_{32}^{(1)} & k_{35}^{(1)} & k_{36}^{(1)} & k_{33}^{(1)} & k_{34}^{(1)} & 0 & 0 \\ k_{41}^{(1)} & k_{42}^{(1)} & k_{45}^{(1)} & k_{46}^{(1)} & k_{43}^{(1)} & k_{44}^{(1)} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{11}^{(2)} & k_{12}^{(2)} & k_{13}^{(2)} & k_{14}^{(2)} & k_{15}^{(2)} & k_{16}^{(2)} \\ 0 & 0 & k_{21}^{(2)} & k_{22}^{(2)} & k_{23}^{(2)} & k_{24}^{(2)} & k_{25}^{(2)} & k_{26}^{(2)} \\ 0 & 0 & k_{31}^{(2)} & k_{32}^{(2)} & k_{33}^{(2)} & k_{34}^{(2)} & k_{35}^{(2)} & k_{36}^{(2)} \\ 0 & 0 & k_{41}^{(2)} & k_{42}^{(2)} & k_{43}^{(2)} & k_{44}^{(2)} & k_{45}^{(2)} & k_{46}^{(2)} \\ 0 & 0 & k_{51}^{(2)} & k_{52}^{(2)} & k_{53}^{(2)} & k_{54}^{(2)} & k_{55}^{(2)} & k_{56}^{(2)} \\ 0 & 0 & k_{61}^{(2)} & k_{62}^{(2)} & k_{63}^{(2)} & k_{64}^{(2)} & k_{65}^{(2)} & k_{66}^{(2)} \end{pmatrix}$$

$$= \begin{pmatrix} k_{11}^{(1)} & k_{12}^{(1)} & k_{15}^{(1)} & k_{16}^{(1)} & k_{13}^{(1)} & k_{14}^{(1)} & 0 & 0 \\ k_{21}^{(1)} & k_{22}^{(1)} & k_{25}^{(1)} & k_{26}^{(1)} & k_{23}^{(1)} & k_{24}^{(1)} & 0 & 0 \\ k_{51}^{(1)} & k_{52}^{(1)} & k_{55}^{(1)} + k_{11}^{(2)} & k_{56}^{(1)} + k_{12}^{(2)} & k_{53}^{(1)} + k_{13}^{(2)} & k_{54}^{(1)} + k_{14}^{(2)} & k_{15}^{(2)} & k_{16}^{(2)} \\ k_{61}^{(1)} & k_{62}^{(1)} & k_{65}^{(1)} + k_{21}^{(2)} & k_{66}^{(1)} + k_{22}^{(2)} & k_{63}^{(1)} + k_{23}^{(2)} & k_{64}^{(1)} + k_{24}^{(2)} & k_{25}^{(2)} & k_{26}^{(2)} \\ k_{31}^{(1)} & k_{32}^{(1)} & k_{35}^{(1)} + k_{31}^{(2)} & k_{36}^{(1)} + k_{32}^{(2)} & k_{33}^{(1)} + k_{33}^{(2)} & k_{34}^{(1)} + k_{34}^{(2)} & k_{35}^{(2)} & k_{36}^{(2)} \\ k_{41}^{(1)} & k_{42}^{(1)} & k_{45}^{(1)} + k_{41}^{(2)} & k_{46}^{(1)} + k_{42}^{(2)} & k_{43}^{(1)} + k_{43}^{(2)} & k_{44}^{(1)} + k_{44}^{(2)} & k_{45}^{(2)} & k_{46}^{(2)} \\ 0 & 0 & k_{51}^{(2)} & k_{52}^{(2)} & k_{53}^{(2)} & k_{54}^{(2)} & k_{55}^{(2)} & k_{56}^{(2)} \\ 0 & 0 & k_{61}^{(2)} & k_{62}^{(2)} & k_{63}^{(2)} & k_{64}^{(2)} & k_{65}^{(2)} & k_{66}^{(2)} \end{pmatrix} \tag{2.84}$$

The same approach are also applied for force vector, because it also share the contribution at

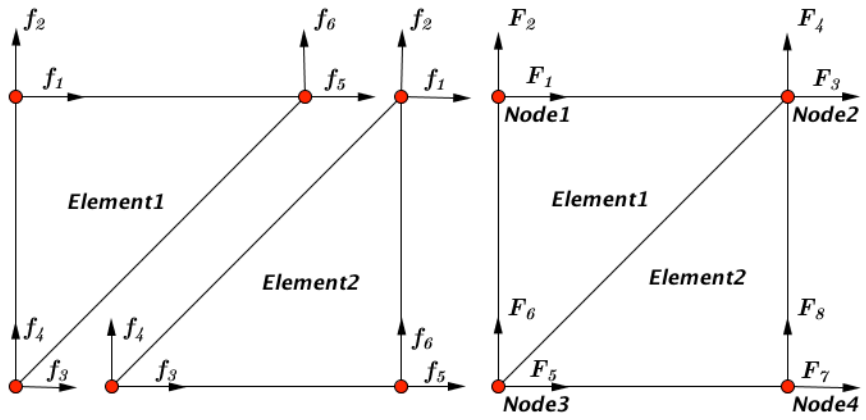


Figure 2.14: Two Tri6 elements in global and local perspective.

degree of freedom. The procedure of assembly are expressed as:

$$F = \begin{pmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_5^{(1)} \\ f_6^{(1)} \\ f_3^{(1)} \\ f_4^{(1)} \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ f_1^{(2)} \\ f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \\ f_5^{(2)} \\ f_6^{(2)} \end{pmatrix} = \begin{pmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_5^{(1)} + f_1^{(2)} \\ f_6^{(1)} + f_2^{(2)} \\ f_3^{(1)} + f_3^{(2)} \\ f_4^{(1)} + f_4^{(2)} \\ f_5^{(2)} \\ f_6^{(2)} \end{pmatrix} \tag{2.85}$$

For assembly of strain and stress, we need to combine the contribution at each node. An extra procedure after combining of strain or stress is to solve the average value by dividing the number of element that share a same node. Because the strain and strain are always jump between elements. The rule for solving averaging strain and stress can be formed as:

$$\epsilon = \sum_{i=1}^{n} \frac{\epsilon_{i,raw}}{n} \qquad \sigma = \sum_{i=1}^{n} \frac{\sigma_{i,raw}}{n} \tag{2.86}$$

$$\epsilon_{raw} = \begin{pmatrix} \epsilon_{1xx}^{(1)} & \epsilon_{1yy}^{(1)} & \epsilon_{1zz}^{(1)} & 2\epsilon_{1yz}^{(1)} & 2\epsilon_{1xz}^{(1)} & 2\epsilon_{1xy}^{(1)} \\ \epsilon_{2xx}^{(1)} & \epsilon_{2yy}^{(1)} & \epsilon_{2zz}^{(1)} & 2\epsilon_{2yz}^{(1)} & 2\epsilon_{2xz}^{(1)} & 2\epsilon_{2xy}^{(1)} \\ \epsilon_{5xx}^{(1)} & \epsilon_{5yy}^{(1)} & \epsilon_{5zz}^{(1)} & 2\epsilon_{5yz}^{(1)} & 2\epsilon_{5xz}^{(1)} & 2\epsilon_{5xy}^{(1)} \\ \epsilon_{6xx}^{(1)} & \epsilon_{6yy}^{(1)} & \epsilon_{6zz}^{(1)} & 2\epsilon_{6yz}^{(1)} & 2\epsilon_{6xz}^{(1)} & 2\epsilon_{6xy}^{(1)} \\ \epsilon_{3xx}^{(1)} & \epsilon_{3yy}^{(1)} & \epsilon_{3zz}^{(1)} & 2\epsilon_{3yz}^{(1)} & 2\epsilon_{3xz}^{(1)} & 2\epsilon_{3xy}^{(1)} \\ \epsilon_{4xx}^{(1)} & \epsilon_{4yy}^{(1)} & \epsilon_{4zz}^{(1)} & 2\epsilon_{4yz}^{(1)} & 2\epsilon_{4xz}^{(1)} & 2\epsilon_{4xy}^{(1)} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \epsilon_{1xx}^{(2)} & \epsilon_{1yy}^{(2)} & \epsilon_{1zz}^{(2)} & 2\epsilon_{1yz}^{(2)} & 2\epsilon_{1xz}^{(2)} & 2\epsilon_{1xy}^{(2)} \\ \epsilon_{2xx}^{(2)} & \epsilon_{2yy}^{(2)} & \epsilon_{2zz}^{(2)} & 2\epsilon_{2yz}^{(2)} & 2\epsilon_{2xz}^{(2)} & 2\epsilon_{2xy}^{(2)} \\ \epsilon_{3xx}^{(2)} & \epsilon_{3yy}^{(2)} & \epsilon_{3zz}^{(2)} & 2\epsilon_{3yz}^{(2)} & 2\epsilon_{3xz}^{(2)} & 2\epsilon_{3xy}^{(2)} \\ \epsilon_{4xx}^{(2)} & \epsilon_{4yy}^{(2)} & \epsilon_{4zz}^{(2)} & 2\epsilon_{4yz}^{(2)} & 2\epsilon_{4xz}^{(2)} & 2\epsilon_{4xy}^{(2)} \\ \epsilon_{5xx}^{(2)} & \epsilon_{5yy}^{(2)} & \epsilon_{5zz}^{(2)} & 2\epsilon_{5yz}^{(2)} & 2\epsilon_{5xz}^{(2)} & 2\epsilon_{5xy}^{(2)} \\ \epsilon_{6xx}^{(2)} & \epsilon_{6yy}^{(2)} & \epsilon_{6zz}^{(2)} & 2\epsilon_{6yz}^{(2)} & 2\epsilon_{6xz}^{(2)} & 2\epsilon_{6xy}^{(2)} \end{pmatrix}$$

$$\epsilon = \begin{pmatrix} \epsilon_{1xx}^{(1)} & \epsilon_{1yy}^{(1)} & \epsilon_{1zz}^{(1)} & 2\epsilon_{1yz}^{(1)} & 2\epsilon_{1xz}^{(1)} & 2\epsilon_{1xy}^{(1)} \\ \epsilon_{2xx}^{(1)} & \epsilon_{2yy}^{(1)} & \epsilon_{2zz}^{(1)} & 2\epsilon_{2yz}^{(1)} & 2\epsilon_{2xz}^{(1)} & 2\epsilon_{2xy}^{(1)} \\ \frac{\epsilon_{5xx}^{(1)}+\epsilon_{1xx}^{(2)}}{2} & \frac{\epsilon_{5yy}^{(1)}+\epsilon_{1yy}^{(2)}}{2} & \frac{\epsilon_{5zz}^{(1)}+\epsilon_{1zz}^{(2)}}{2} & \frac{2\epsilon_{5yz}^{(1)}+2\epsilon_{1yz}^{(2)}}{2} & \frac{2\epsilon_{5xz}^{(1)}+2\epsilon_{1xz}^{(2)}}{2} & \frac{2\epsilon_{5xy}^{(1)}+2\epsilon_{1xy}^{(2)}}{2} \\ \frac{\epsilon_{6xx}^{(1)}+\epsilon_{2xx}^{(2)}}{2} & \frac{\epsilon_{6yy}^{(1)}+\epsilon_{2yy}^{(2)}}{2} & \frac{\epsilon_{6zz}^{(1)}+\epsilon_{2zz}^{(2)}}{2} & \frac{2\epsilon_{6yz}^{(1)}+2\epsilon_{2yz}^{(2)}}{2} & \frac{2\epsilon_{6xz}^{(1)}+2\epsilon_{2xz}^{(2)}}{2} & \frac{2\epsilon_{6xy}^{(1)}+2\epsilon_{2xy}^{(2)}}{2} \\ \frac{\epsilon_{3xx}^{(1)}+\epsilon_{3xx}^{(2)}}{2} & \frac{\epsilon_{3yy}^{(1)}+\epsilon_{3yy}^{(2)}}{2} & \frac{\epsilon_{3zz}^{(1)}+\epsilon_{3zz}^{(2)}}{2} & \frac{2\epsilon_{3yz}^{(1)}+2\epsilon_{3yz}^{(2)}}{2} & \frac{2\epsilon_{3xz}^{(1)}+2\epsilon_{3xz}^{(2)}}{2} & \frac{2\epsilon_{3xy}^{(1)}+2\epsilon_{3xy}^{(2)}}{2} \\ \frac{\epsilon_{4xx}^{(1)}+\epsilon_{4xx}^{(2)}}{2} & \frac{\epsilon_{4yy}^{(1)}+\epsilon_{4yy}^{(2)}}{2} & \frac{\epsilon_{4zz}^{(1)}+\epsilon_{4zz}^{(2)}}{2} & \frac{2\epsilon_{4yz}^{(1)}+2\epsilon_{4yz}^{(2)}}{2} & \frac{2\epsilon_{4xz}^{(1)}+2\epsilon_{4xz}^{(2)}}{2} & \frac{2\epsilon_{4xy}^{(1)}+2\epsilon_{4xy}^{(2)}}{2} \\ \epsilon_{5xx}^{(2)} & \epsilon_{5yy}^{(2)} & \epsilon_{5zz}^{(2)} & 2\epsilon_{5yz}^{(2)} & 2\epsilon_{5xz}^{(2)} & 2\epsilon_{5xy}^{(2)} \\ \epsilon_{6xx}^{(2)} & \epsilon_{6yy}^{(2)} & \epsilon_{6zz}^{(2)} & 2\epsilon_{6yz}^{(2)} & 2\epsilon_{6xz}^{(2)} & 2\epsilon_{6xy}^{(2)} \end{pmatrix} \tag{2.87}$$

$$\sigma_{raw} = \begin{pmatrix} \sigma^{(1)}_{1xx} & \sigma^{(1)}_{1yy} & \sigma^{(1)}_{1zz} & \sigma^{(1)}_{1yz} & \sigma^{(1)}_{1xz} & \sigma^{(1)}_{1xy} \\ \sigma^{(1)}_{2xx} & \sigma^{(1)}_{2yy} & \sigma^{(1)}_{2zz} & \sigma^{(1)}_{2yz} & \sigma^{(1)}_{2xz} & f^{(1)}_{2xy} \\ \sigma^{(1)}_{5xx} & \sigma^{(1)}_{5yy} & \sigma^{(1)}_{5zz} & \sigma^{(1)}_{5yz} & \sigma^{(1)}_{5xz} & \sigma^{(1)}_{5xy} \\ \sigma^{(1)}_{6xx} & \sigma^{(1)}_{6yy} & \sigma^{(1)}_{6zz} & \sigma^{(1)}_{6yz} & \sigma^{(1)}_{6xz} & \sigma^{(1)}_{6xy} \\ \sigma^{(1)}_{3xx} & \sigma^{(1)}_{3yy} & \sigma^{(1)}_{3zz} & \sigma^{(1)}_{3yz} & \sigma^{(1)}_{3xz} & \sigma^{(1)}_{3xy} \\ \sigma^{(1)}_{4xx} & \sigma^{(1)}_{4yy} & \sigma^{(1)}_{4zz} & \sigma^{(1)}_{4yz} & \sigma^{(1)}_{4xz} & \sigma^{(1)}_{4xy} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \sigma^{(2)}_{1xx} & \sigma^{(2)}_{1yy} & \sigma^{(2)}_{1zz} & \sigma^{(2)}_{1yz} & \sigma^{(2)}_{1xz} & \sigma^{(2)}_{1xy} \\ \sigma^{(2)}_{2xx} & \sigma^{(2)}_{2yy} & \sigma^{(2)}_{2zz} & \sigma^{(2)}_{2yz} & \sigma^{(2)}_{2xz} & \sigma^{(2)}_{2xy} \\ \sigma^{(2)}_{3xx} & \sigma^{(2)}_{3yy} & \sigma^{(2)}_{3zz} & \sigma^{(2)}_{3yz} & \sigma^{(2)}_{3xz} & \sigma^{(2)}_{3xy} \\ \sigma^{(2)}_{4xx} & \sigma^{(2)}_{4yy} & \sigma^{(2)}_{4zz} & \sigma^{(2)}_{4yz} & \sigma^{(2)}_{4xz} & \sigma^{(2)}_{4xy} \\ \sigma^{(2)}_{5xx} & \sigma^{(2)}_{5yy} & \sigma^{(2)}_{5zz} & \sigma^{(2)}_{5yz} & \sigma^{(2)}_{5xz} & \sigma^{(2)}_{5xy} \\ \sigma^{(2)}_{6xx} & \sigma^{(2)}_{6yy} & \sigma^{(2)}_{6zz} & \sigma^{(2)}_{6yz} & \sigma^{(2)}_{6xz} & \sigma^{(2)}_{6xy} \end{pmatrix}$$

$$\sigma = \begin{pmatrix} \sigma^{(1)}_{1xx} & \sigma^{(1)}_{1yy} & \sigma^{(1)}_{1zz} & \sigma^{(1)}_{1yz} & \sigma^{(1)}_{1xz} & \sigma^{(1)}_{1xy} \\ \sigma^{(1)}_{2xx} & \sigma^{(1)}_{2yy} & \sigma^{(1)}_{2zz} & \sigma^{(1)}_{2yz} & \sigma^{(1)}_{2xz} & f^{(1)}_{2xy} \\ \frac{\sigma^{(1)}_{5xx}+\sigma^{(2)}_{1xx}}{2} & \frac{\sigma^{(1)}_{5yy}+\sigma^{(2)}_{1yy}}{2} & \frac{\sigma^{(1)}_{5zz}+\sigma^{(2)}_{1zz}}{2} & \frac{\sigma^{(1)}_{5yz}+\sigma^{(2)}_{1yz}}{2} & \frac{\sigma^{(1)}_{5xz}+\sigma^{(2)}_{1xz}}{2} & \frac{\sigma^{(1)}_{5xy}+\sigma^{(2)}_{1xy}}{2} \\ \frac{\sigma^{(1)}_{6xx}+\sigma^{(2)}_{2xx}}{2} & \frac{\sigma^{(1)}_{6yy}+\sigma^{(2)}_{2yy}}{2} & \frac{\sigma^{(1)}_{6zz}+\sigma^{(2)}_{2zz}}{2} & \frac{\sigma^{(1)}_{6yz}+\sigma^{(2)}_{2yz}}{2} & \frac{\sigma^{(1)}_{6xz}+\sigma^{(2)}_{2xz}}{2} & \frac{\sigma^{(1)}_{6xy}+\sigma^{(2)}_{2xy}}{2} \\ \frac{\sigma^{(1)}_{3xx}+\sigma^{(2)}_{3xx}}{2} & \frac{\sigma^{(1)}_{3yy}+\sigma^{(2)}_{3yy}}{2} & \frac{\sigma^{(1)}_{3zz}+\sigma^{(2)}_{3zz}}{2} & \frac{\sigma^{(1)}_{3yz}+\sigma^{(2)}_{3yz}}{2} & \frac{\sigma^{(1)}_{3xz}+\sigma^{(2)}_{3xz}}{2} & \frac{\sigma^{(1)}_{3xy}+\sigma^{(2)}_{3xy}}{2} \\ \frac{\sigma^{(1)}_{4xx}+\sigma^{(2)}_{4xx}}{2} & \frac{\sigma^{(1)}_{4yy}+\sigma^{(2)}_{4yy}}{2} & \frac{\sigma^{(1)}_{4zz}+\sigma^{(2)}_{4zz}}{2} & \frac{\sigma^{(1)}_{4yz}+\sigma^{(2)}_{4yz}}{2} & \frac{\sigma^{(1)}_{4xz}+\sigma^{(2)}_{4xz}}{2} & \frac{\sigma^{(1)}_{4xy}+\sigma^{(2)}_{4xy}}{2} \\ \sigma^{(2)}_{5xx} & \sigma^{(2)}_{5yy} & \sigma^{(2)}_{5zz} & \sigma^{(2)}_{5yz} & \sigma^{(2)}_{5xz} & \sigma^{(2)}_{5xy} \\ \sigma^{(2)}_{6xx} & \sigma^{(2)}_{6yy} & \sigma^{(2)}_{6zz} & \sigma^{(2)}_{6yz} & \sigma^{(2)}_{6xz} & \sigma^{(2)}_{6xy} \end{pmatrix} \tag{2.88}$$

Chapter 3

Programming Implementation

3.1 Pre-Processing

The purpose of Pre-Processing is to build a numerical model and export as two lists- node list and element list. The data of model are ultimately imported into AMfe Toolbox. ANSYS Parametric Design Language (APDL) code is the tool to create a meshed geometry in ANSYS. A simple meshed geometry are shown in Figure 3.1. Task step for building model contains a series of related path. An example of a path step is:

Geometry ⇒ Element Type ⇒ Material ⇒ Mesh ⇒ Boundary Condition

After building the model, numerical data can be exported as node list and element list. Node list contains coordinate all the nodes of geometry. Element list are formed with the attributes of each elements and the index of node with belongs to element. The element attributes are assigned to different parts of model by 'pointing' to the appropriate entries in the element list. The pointers are simply a set of reference numbers that include the element index (ELEM), material number (MAT), element type number (TYP), real constant set number (REAL), a coordinate system number (ESY) and section ID number (SEC). Connecting with the example in Figure 3.1, both lists are depicted in Table 3.1 and Table 3.2. The numerical information of both lists are captured with regular expression approach and stored in AMfe Toolbox. For different data it is necessary to store in specific data type. Coordinate of each node and the index of nodes from elements are stored as Numpy array. NumPy is the fundamental package for scientific computing with Python. The advantage of Numpy for storage of node and element information depends on its efficient multi-dimensional container of generic data. It contains a powerful N-dimensional array object and sophisticated (broadcasting) functions. Moreover, it is possible for integrating Fortran code. General attributes of element can be stored as a efficient form-Pandas DataFrames. Pandas is a powerful data analysis toolkit, which provides fast, flexible, and expressive data structures designed to make working with relational or labeled data both easy and intuitive. Pandas DataFrames is a two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). Arithmetic operations align on both row and column labels. This format can be thought of as a dictionary-like container for Series objects.

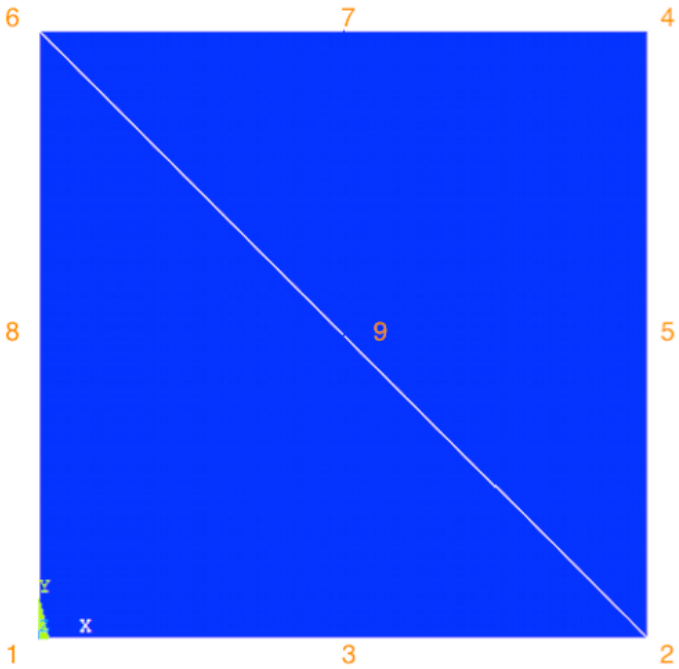


Figure 3.1: A simple meshed geometry

Table 3.1: Node list exported from ANSYS

NODE	X	Y	Z	THXY	THYZ	THZX
1	0.0000	0.0000	0.0000	0.00	0.00	0.00
2	50.000	0.0000	0.0000	0.00	0.00	0.00
3	25.000	0.0000	0.0000	0.00	0.00	0.00
4	50.000	50.000	0.0000	0.00	0.00	0.00
5	50.000	25.000	0.0000	0.00	0.00	0.00
6	0.0000	50.000	0.0000	0.00	0.00	0.00
7	25.000	50.000	0.0000	0.00	0.00	0.00
8	0.0000	25.000	0.0000	0.00	0.00	0.00
9	25.000	25.000	0.0000	0.00	0.00	0.00

Table 3.2: Element list exported from ANSYS

ELEM	MAT	TYP	REL	ESY	SEC	Node	Node	Node	Node	Node	Node
1	1	1	1	0	1	6	2	4	9	5	7
2	1	1	1	0	1	6	1	2	8	3	9

Chapter 4

Element Type Test

4.1 Unit Testing with Python

Automated unit testing is the best way to determine if the element type works well. Unit test is the batteries-included test module in the Python standard library. The motivation for writing a unit testing is to evaluate the quality of each element type, and try to find reason for failed test. The element type to be tested are: Tri3, Tri6, Quad4, Quad8, Tet4 and Tet10. Strain and stress components are of interest to test. As mentioned before, the numerical solution to a mechanical problem is only an approximate solution. There is an error between the analytical solution and FEM solution for sure. Here we do not have an analytical solution to compare. The component from ANSYS will be compared to the results that calculated from AMfe Toolbox. Although the solution from ANSYS is not the analytical solution, it could still help us to check the quality of element type in AMfe Toolbox. For a compare test we need to define the absolute error and relative error. Absolute error is the magnitude of the difference between actual value and approximate value. It can be defined as:

$$e_{abs} = \|X_{actual} - X_{fem}\| \tag{4.1}$$

The relative error is the absolute error relative to the actual value. It can be defined as:

$$e_{rel} = \frac{\|X_{actual} - X_{fem}\|}{X_{actual}} \tag{4.2}$$

It is noteworthy that errors of approximation or rounding and truncation are introduced by using numerical methods or algorithms and computing with finite precision. Truncation error describes the errors, which occurs because some series (finite or infinite) is truncated to a fewer number of terms. Such errors are virtually algorithmic errors. Roundoff error occurs because of the computing tool's inability to handle certain numbers[Dukkipati 2010]. It is hard to set one tolerance for all engineering problems. The tolerance for error can be various in different situation. One thing we can do here is to set the tolerance with different value and evaluate the highest tolerance for each element type. The whole tests of all elements type are described in Table 4.1. It is obvious that only Tri3 shows a good performance and Quad4 has also a relative good quality. It is necessary to investigate what is wrong with the other element type.

Tolerance Parameter	Tri3	Tri6	Quad4	Quad8	Tet4	Tet10
$e_{abs} = 10^{-10}; e_{rel} = 10^{-4}$	Fail	Fail	Fail	Fail	Fail	Fail
$e_{abs} = 10^{-10}; e_{rel} = 5 \times 10^{-4}$	Pass	Fail	Fail	Fail	Fail	Fail
$e_{abs} = 10^{-10}; e_{rel} = 10^{-3}$	Pass	Fail	Pass	Fail	Fail	Fail

4.2 The Patch Test

After testing the quality of each element type, we need to know if the finite element program have the algorithmic procedure correctly. This is generally not easy to verify if the modelling is too complex. The patch test is an useful technique to check the performance of element type that is being tested. The basic idea of the patch test is to apply special boundary condition to keep a constant strain/stress state. It is to simplify a complex problem to a simple one, which you already know the correct numerical results.
The procedure of displacement patch test is as follows: create a simple geometry that consist of one or several element; pick a patch and apply the displacement at exterior nodes, the prescribed displacement field will be set as $u_x = x, u_y = 0$. The reason that the displacement field only applies at exterior nodes is displacements are related to the background continuum. The last step is to verify if the patch has a constant strain state. Now we take Tri3 as an example to implement the displacement patch test. We create a simple rectangle geometry that consist of two Tri3 element in ANSYS, and the prescribed displacement field is depicted in Figure 4.1. Strain state can be viewed in Figure 4.2 in ANSYS. As expected, $\epsilon_{xx} = 0$. So we have built the model for a patch test successfully. Now it is time to check if AMfe Toolbox has a same constant strain state. We export the node, element and displacement data from ANSYS and import all of them into AMfe Toolbox. After computing of strain and strtess in AMfe, the contour plot of strain that shown in ParaView as Figure 4.2 right side. The plot from AMfe Toolbox illustrate the same result $\epsilon_{xx} = 0$. It proves that the displacement patch test has been passed. It is worth noting that the patch test has only two result: success or failure. There is no a state which like pass the patch test 90 percent. Another element type like Tri3, Quad4, Quad8, Tet4 and Tet10 have also passed the displacement patch test. Taking into account the limited space of this thesis, the details are no longer listed here. From the patch test we can not find any problem why Quad8, Tri6, Tet4 and Tet10 did not passed the unit test. This is still a issue to discuss later.

4.3 Complex Model Test

The patch test is only investigated for simple model, it is of interest that how these element type behave with a complex model. Now we create a relative complex model for 2-D element and 3-D element, respectively. The motivation is to compare the strain and stress contour plot of each element type from ANSYS and AMfe Toolbox. For 2-D model we pick a failed element type Tri6 as example

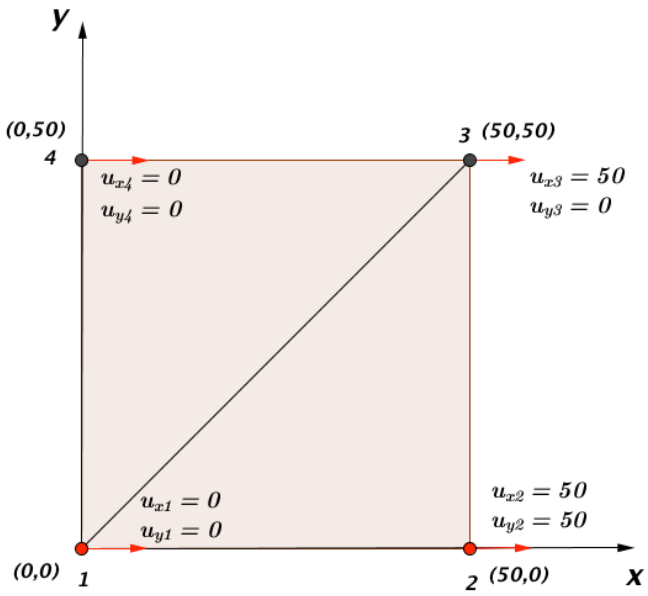


Figure 4.1: The displacement patch test with Tri6 element

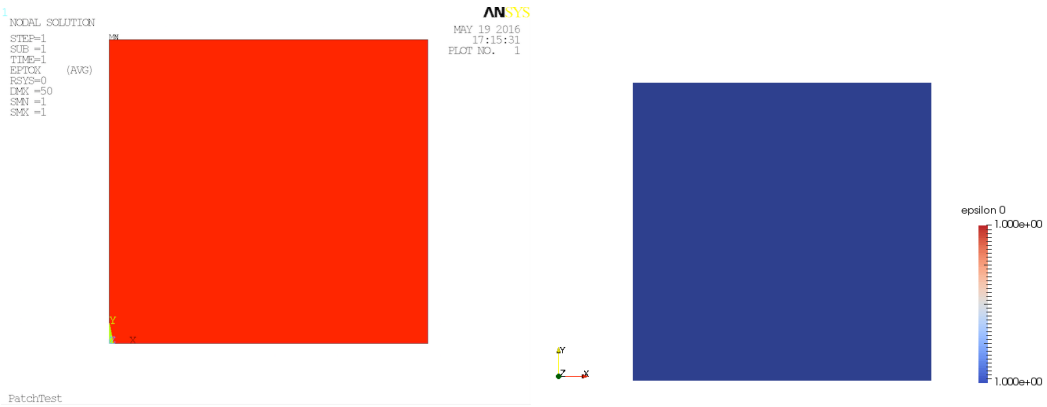


Figure 4.2: left: Contour plot of ϵ_{xx} from ANSYS (SMN: minimal value; SMX: maximal value); right: Contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

For 3-D model we pick another failed element type Tet10

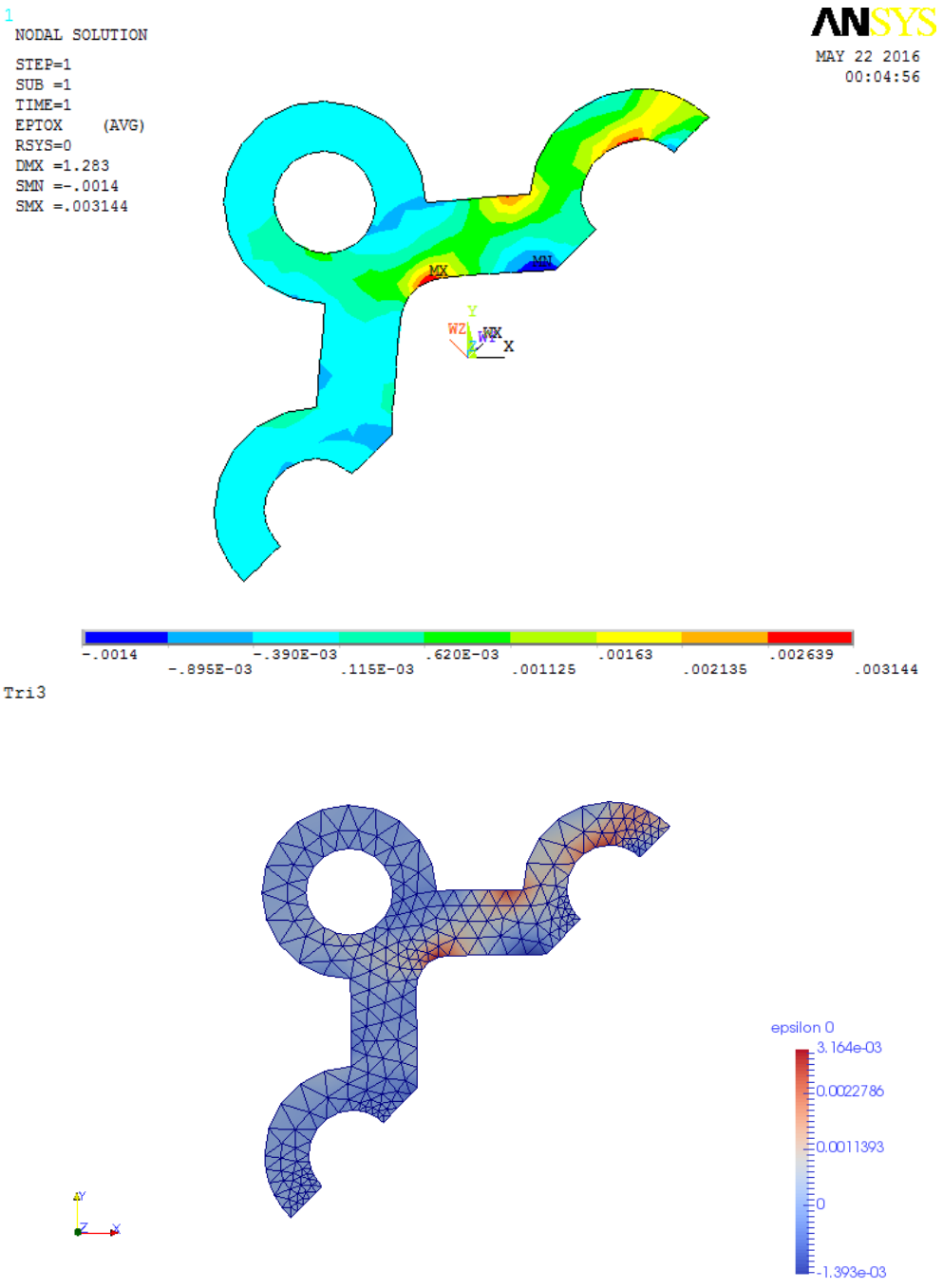


Figure 4.3: Mesh with Tri3. upper: contour plot of ϵ_{xx} from ANSYS; lower: contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

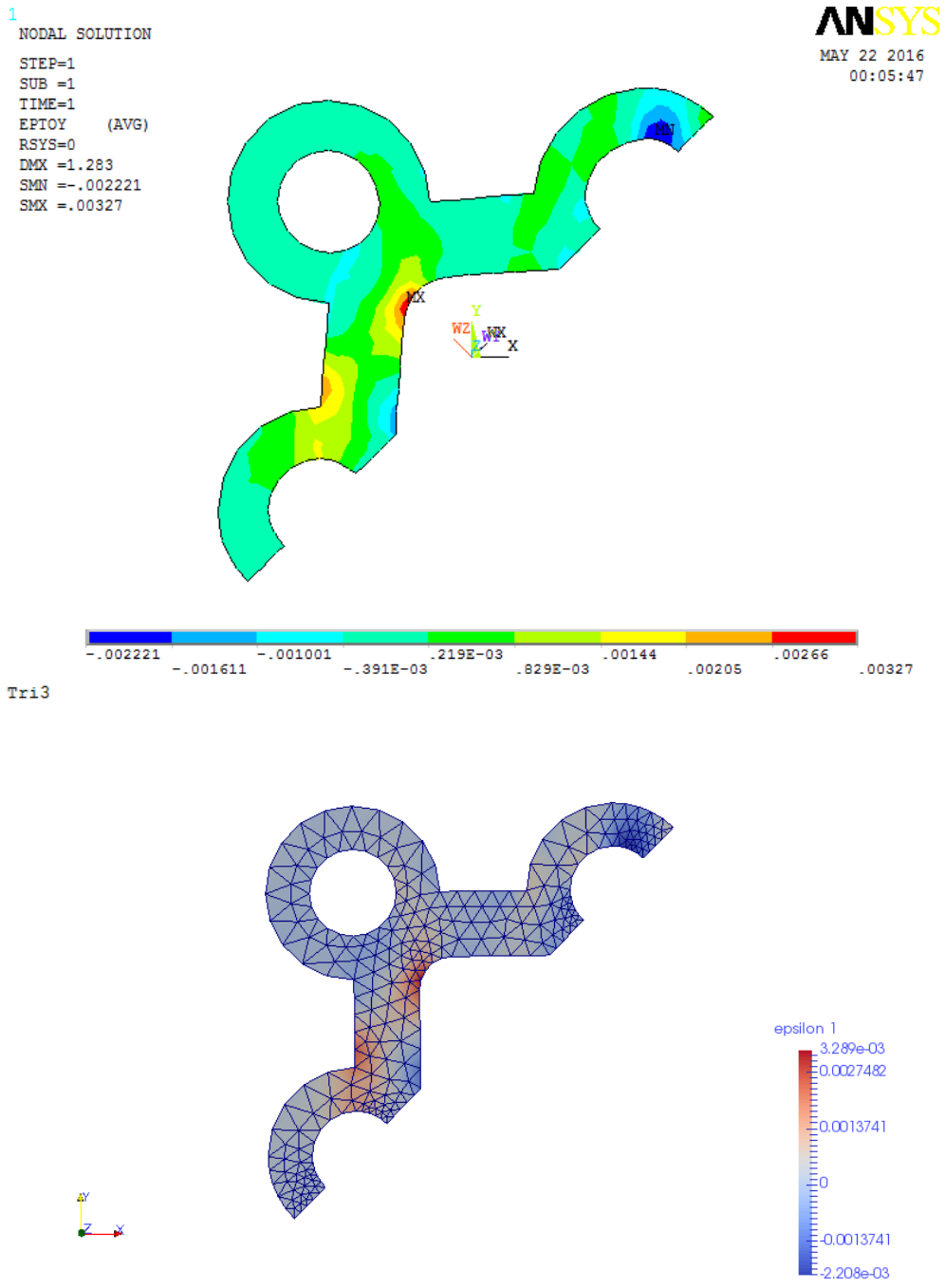


Figure 4.4: Mesh with Tri3. upper: contour plot of ϵ_{yy} from ANSYS; lower: contour plot of ϵ_{yy} calculated in AMfe, demonstrate in ParaView

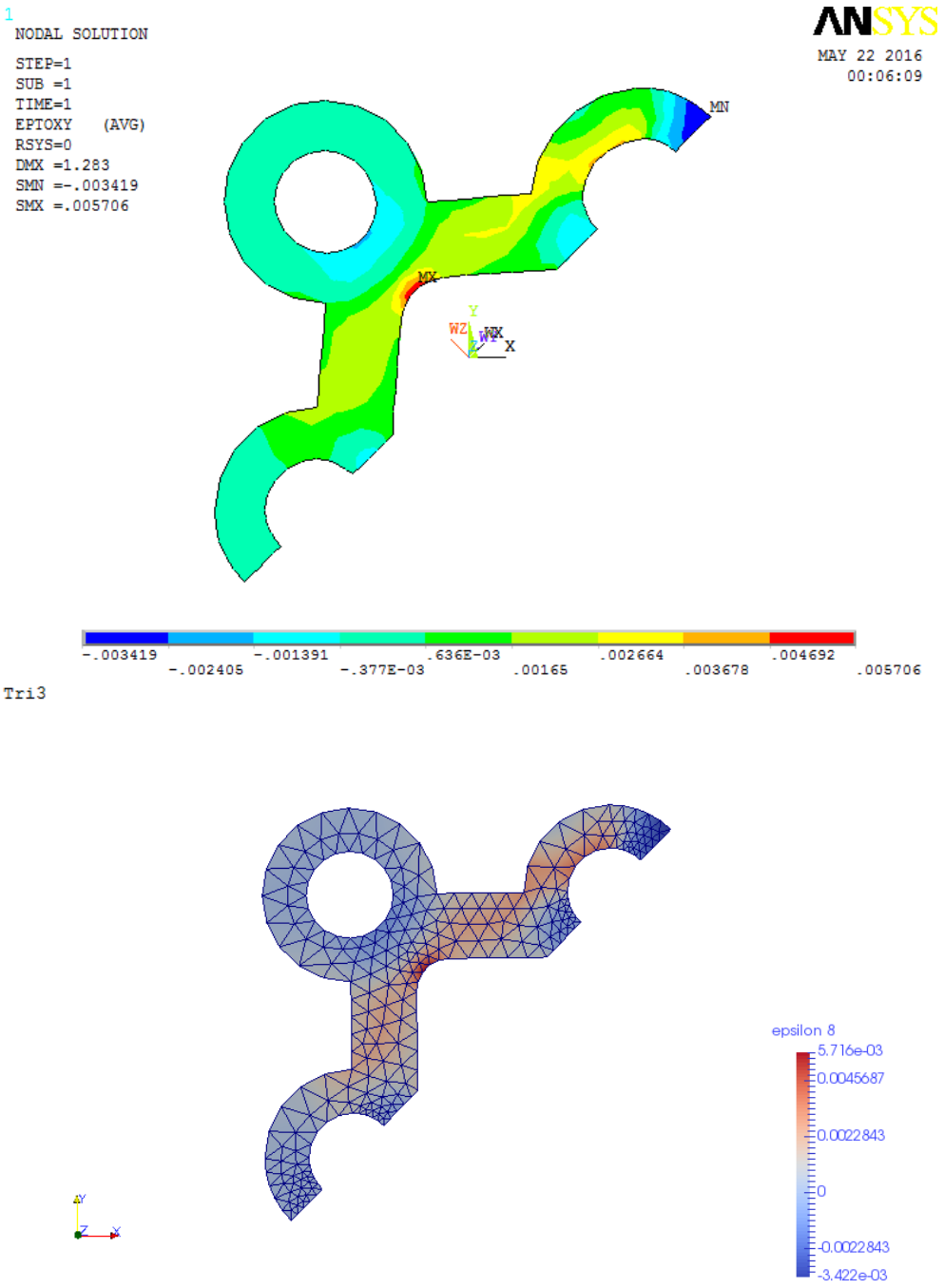


Figure 4.5: Mesh with Tri3. upper: contour plot of ϵ_{xy} from ANSYS; lower: contour plot of ϵ_{xy} calculated in AMfe, demonstrate in ParaView

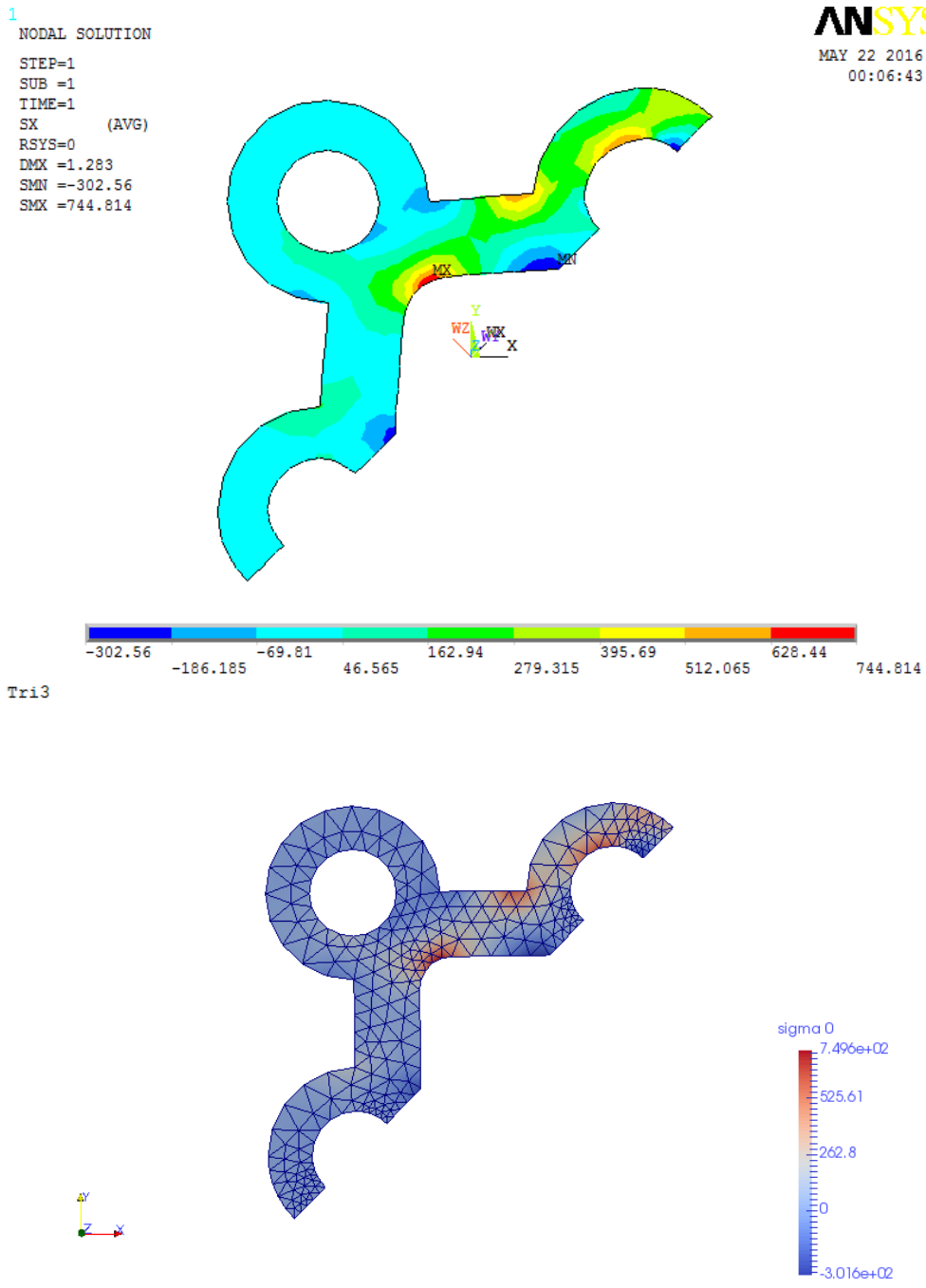


Figure 4.6: Mesh with Tri3. upper: contour plot of σ_{xx} from ANSYS; lower: contour plot of σ_{xx} calculated in AMfe, demonstrate in ParaView

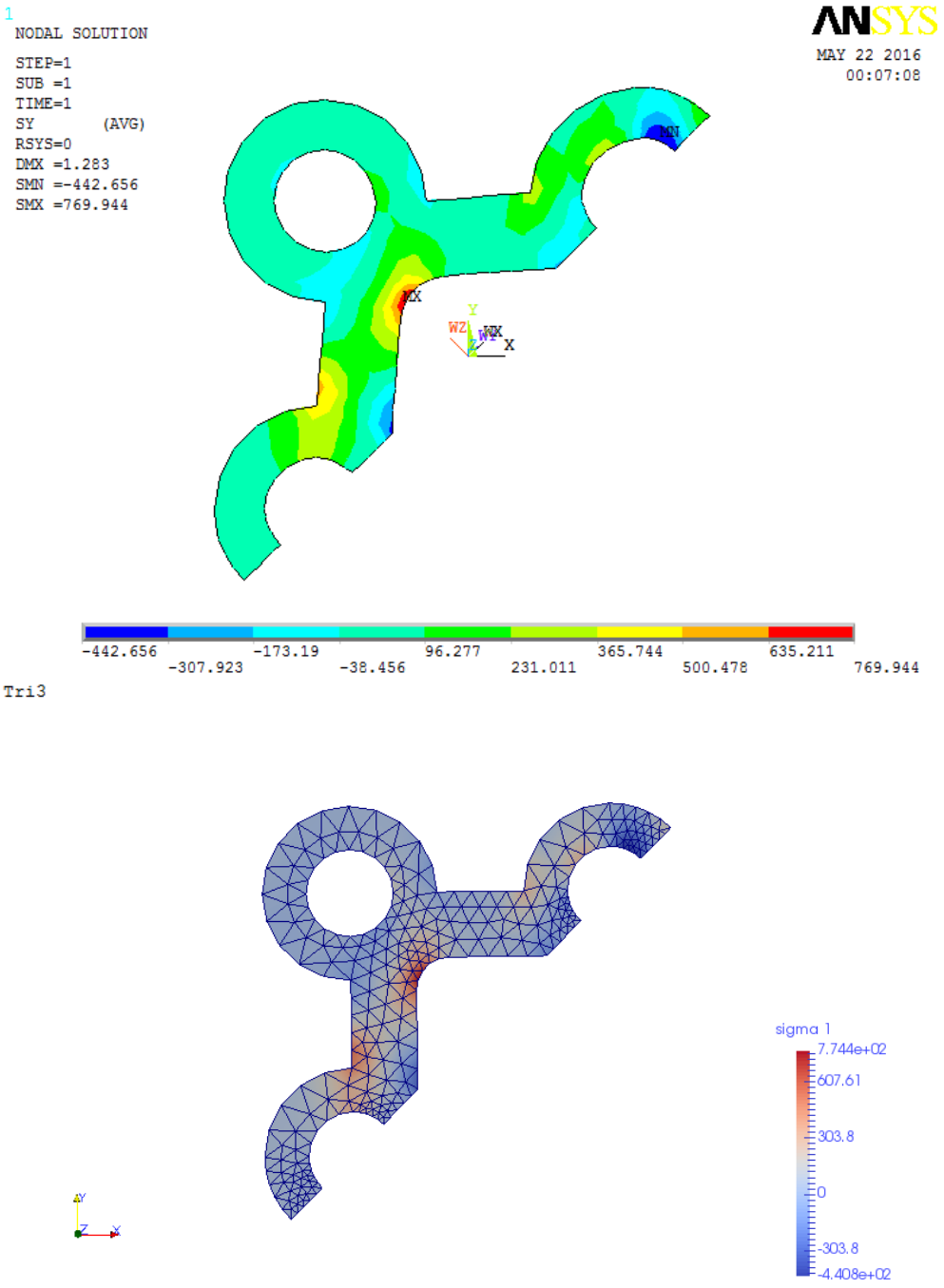


Figure 4.7: Mesh with Tri3. upper: contour plot of σ_{yy} from ANSYS; lower: contour plot of σ_{yy} calculated in AMfe, demonstrate in ParaView

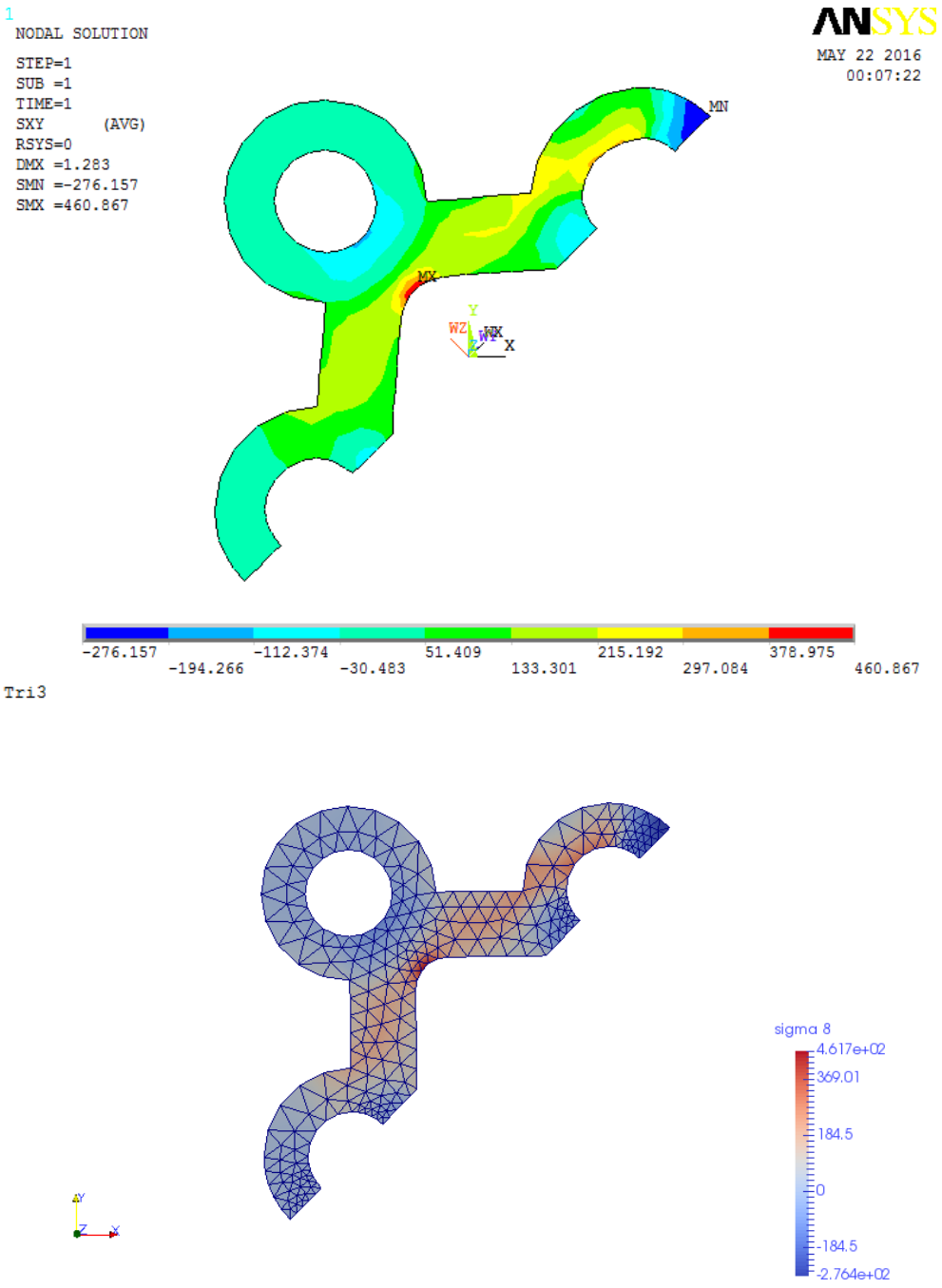


Figure 4.8: Mesh with Tri3. upper: contour plot of σ_{xy} from ANSYS; lower: contour plot of σ_{xy} calculated in AMfe, demonstrate in ParaView

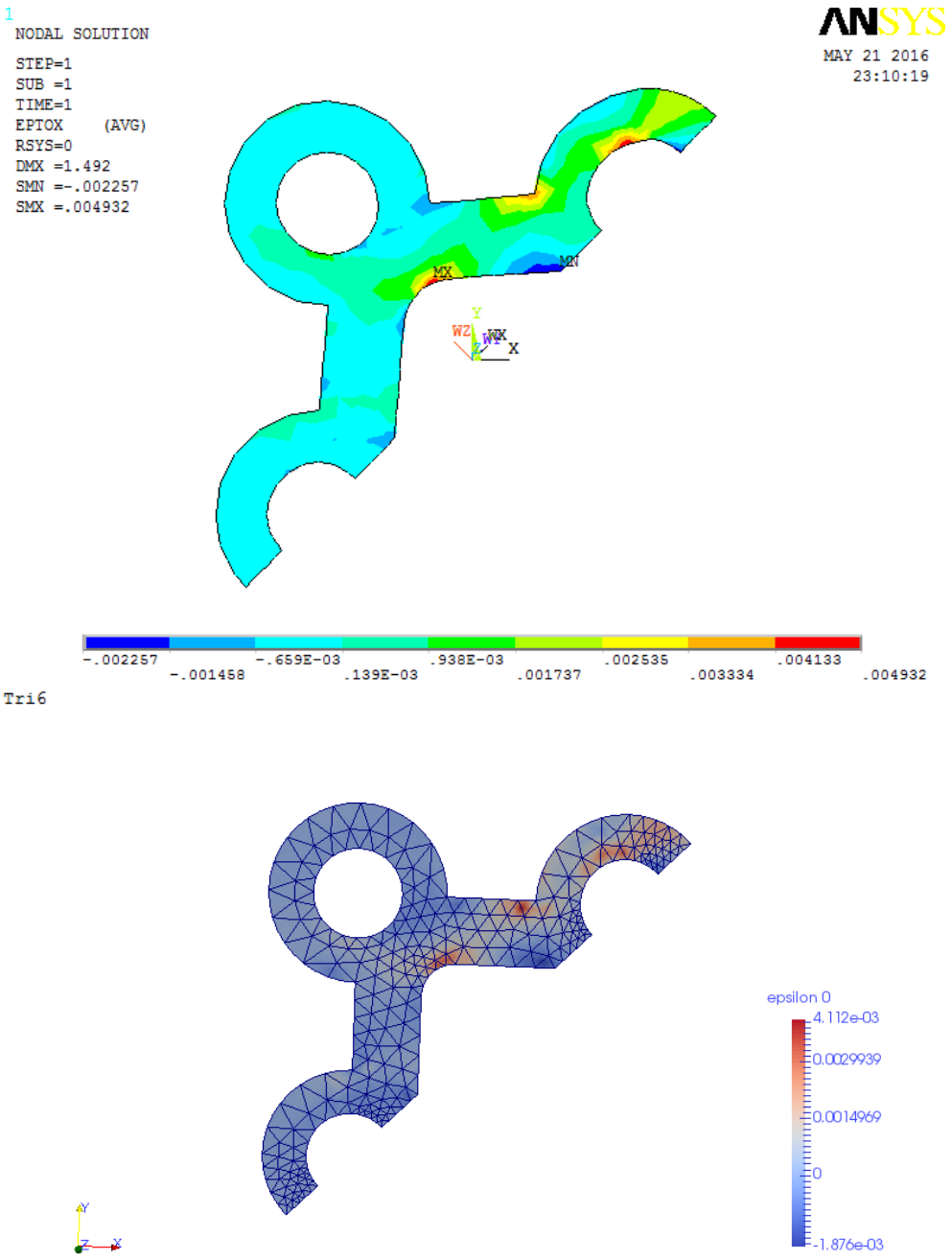


Figure 4.9: Mesh with Tri6. upper: contour plot of ϵ_{xx} from ANSYS; lower: contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

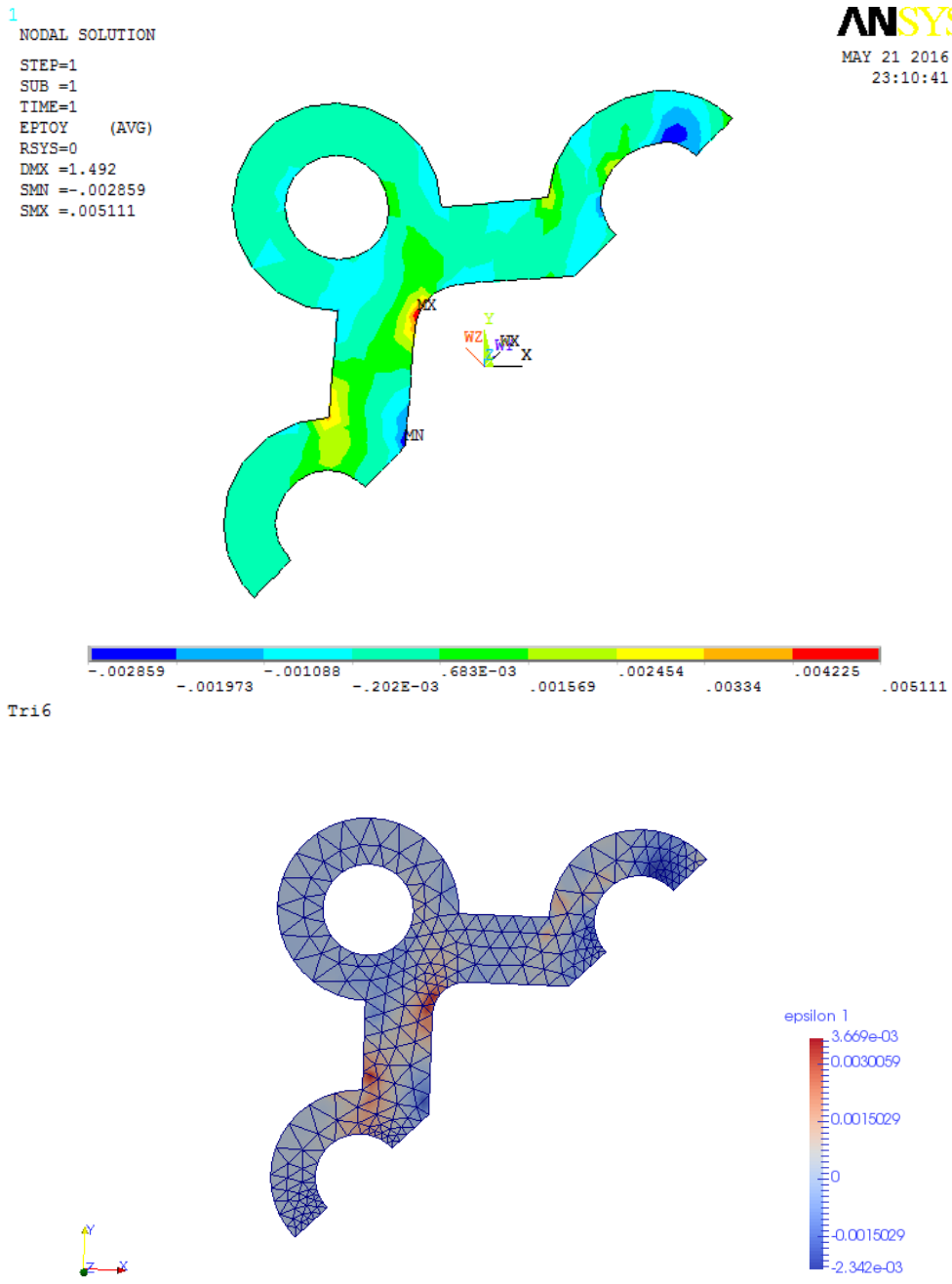


Figure 4.10: Mesh with Tri6. upper: contour plot of ϵ_{yy} from ANSYS; lower: contour plot of ϵ_{yy} calculated in AMfe, demonstrate in ParaView

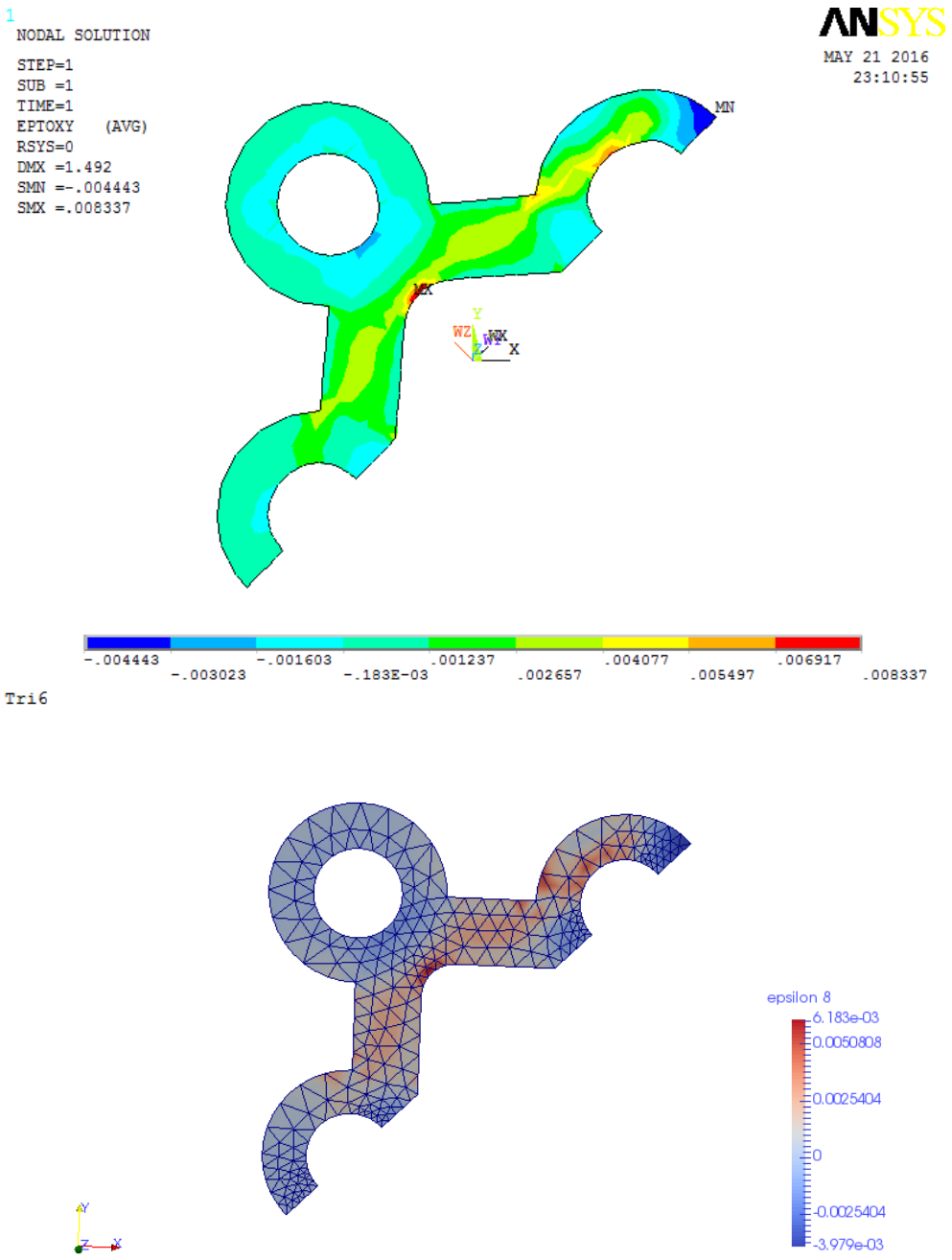


Figure 4.11: Mesh with Tri6. upper: contour plot of ϵ_{xy} from ANSYS; lower: contour plot of ϵ_{xy} calculated in AMfe, demonstrate in ParaView

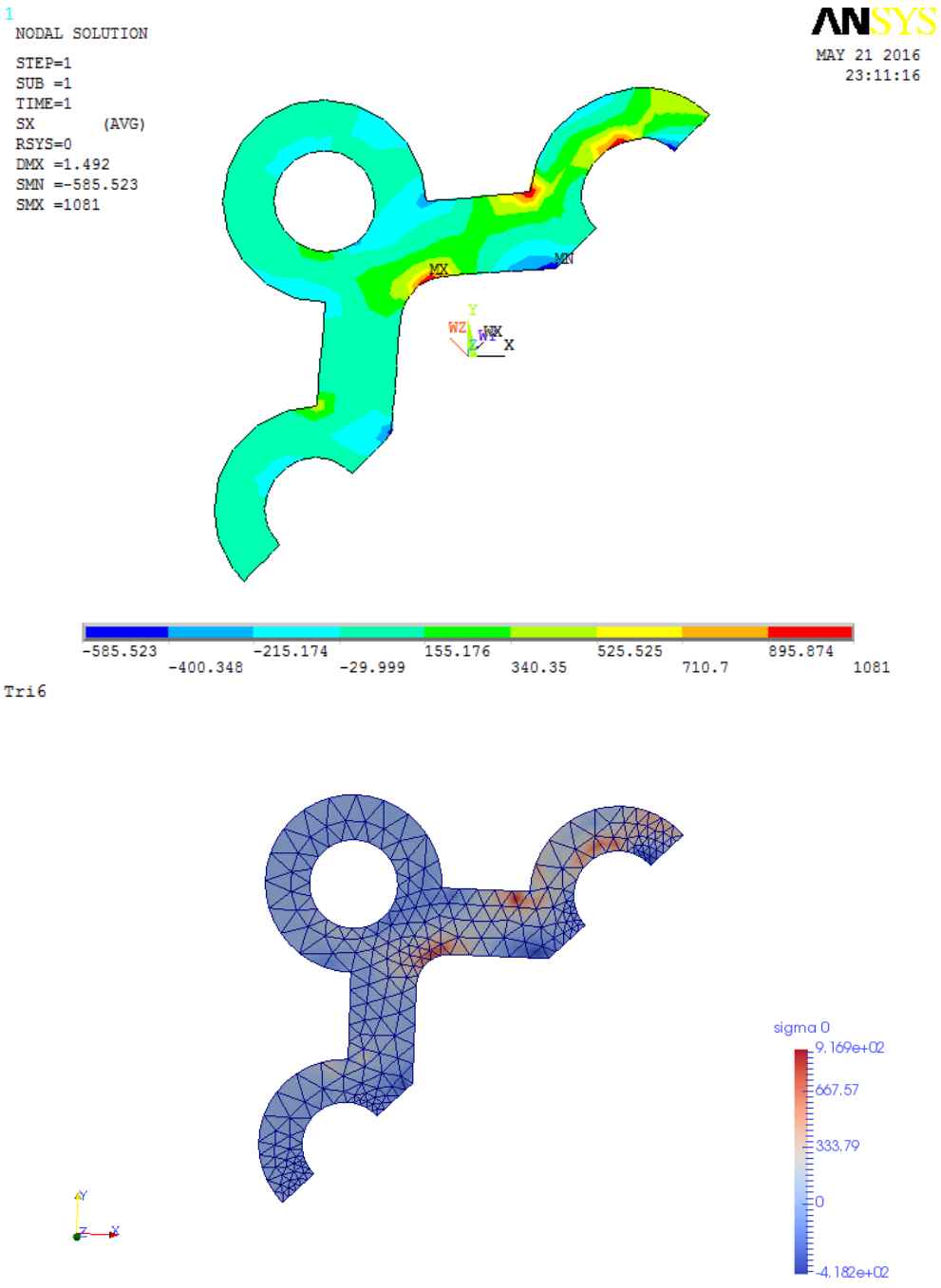


Figure 4.12: Mesh with Tri6. upper: contour plot of σ_{xx} from ANSYS; lower: contour plot of σ_{xx} calculated in AMfe, demonstrate in ParaView

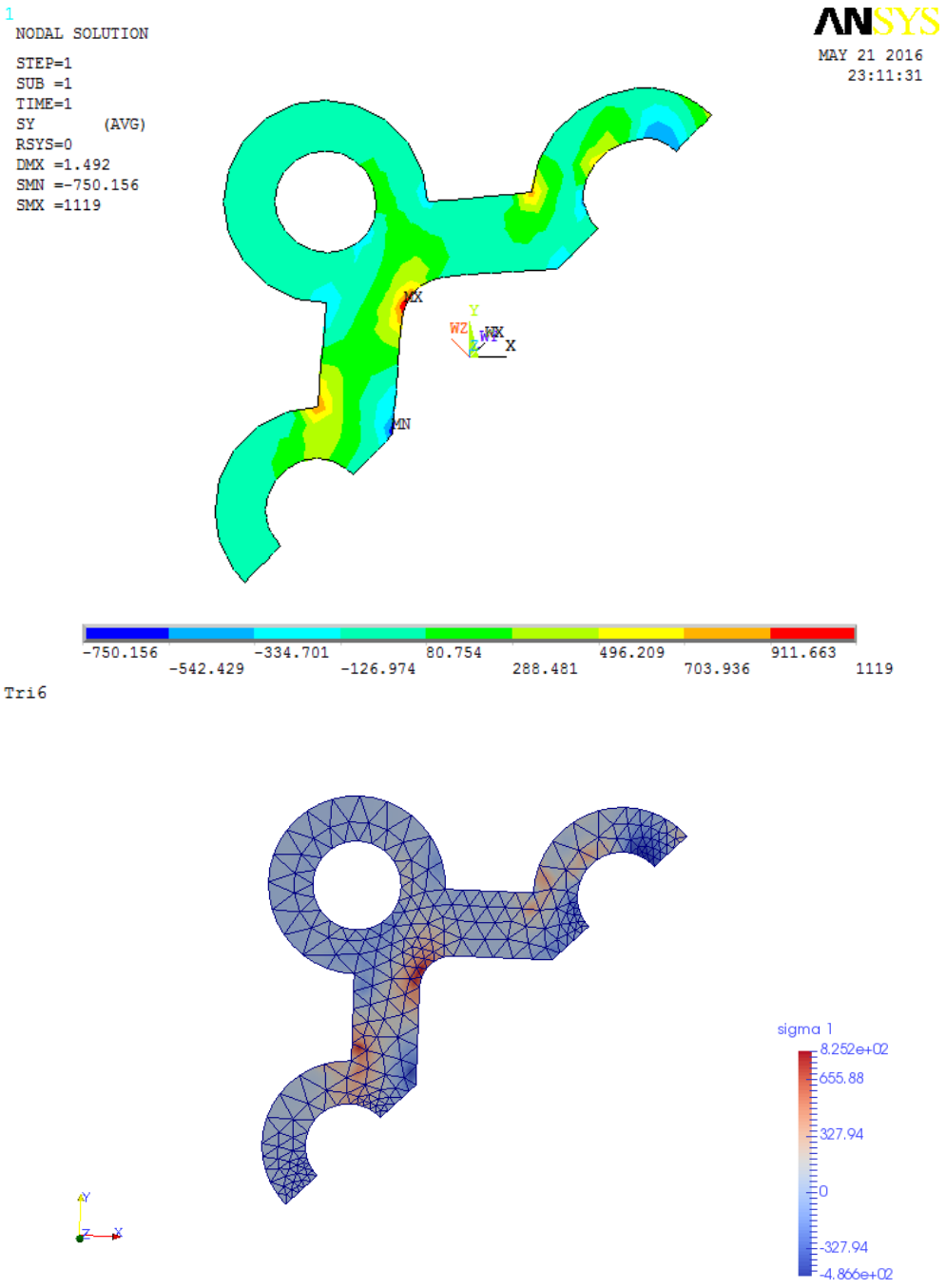
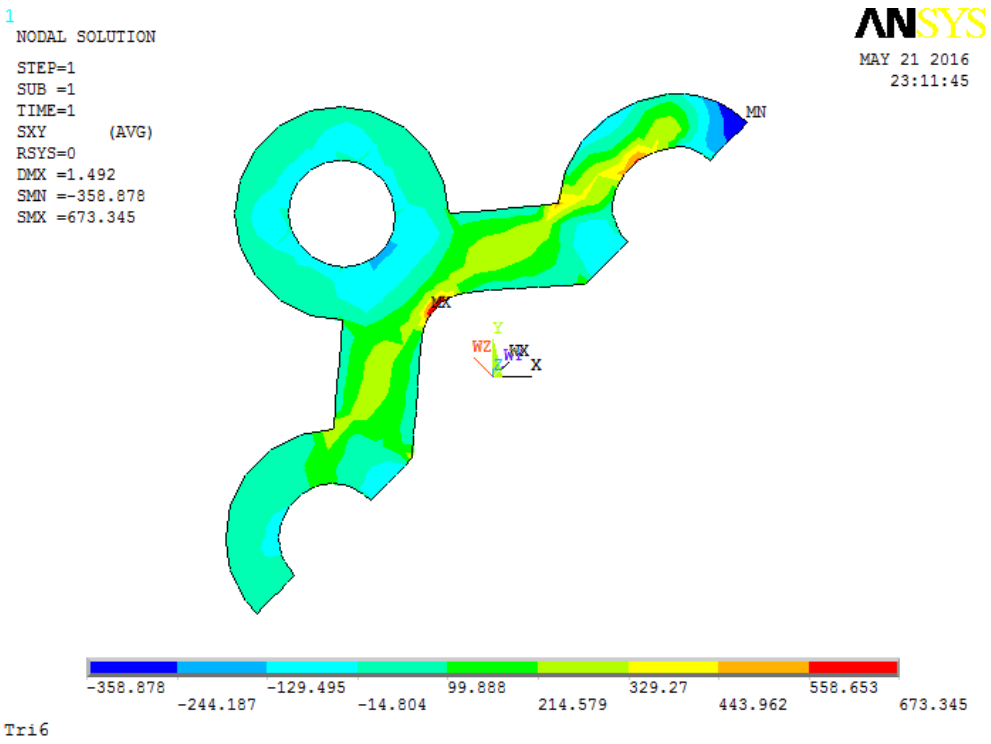


Figure 4.13: Mesh with Tri6. upper: contour plot of σ_{yy} from ANSYS; lower: contour plot of σ_{yy} calculated in AMfe, demonstrate in ParaView



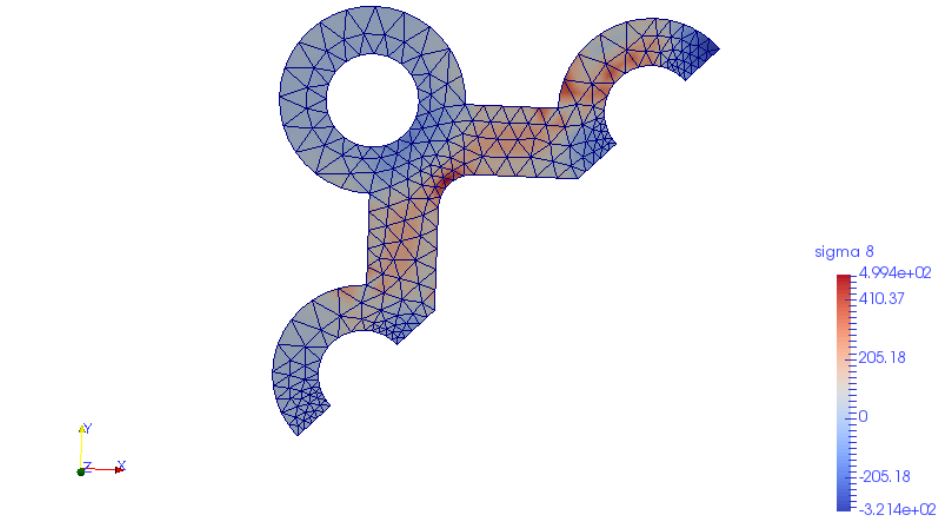


Figure 4.14: Mesh with Tri6. upper: contour plot of σ_{xy} from ANSYS; lower: contour plot of σ_{xy} calculated in AMfe, demonstrate in ParaView

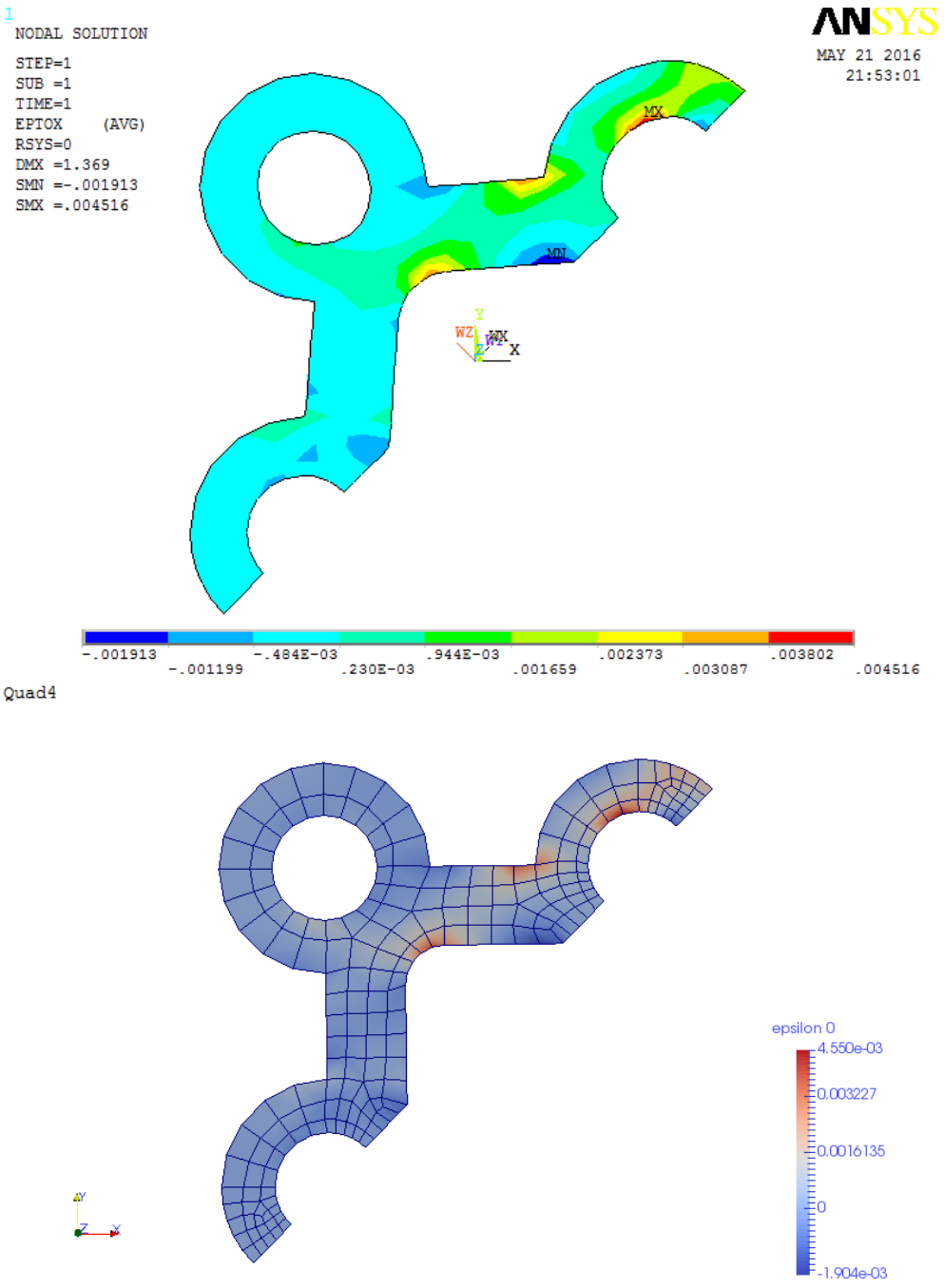


Figure 4.15: Mesh with Quad4. upper: contour plot of ϵ_{xx} from ANSYS; lower: contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

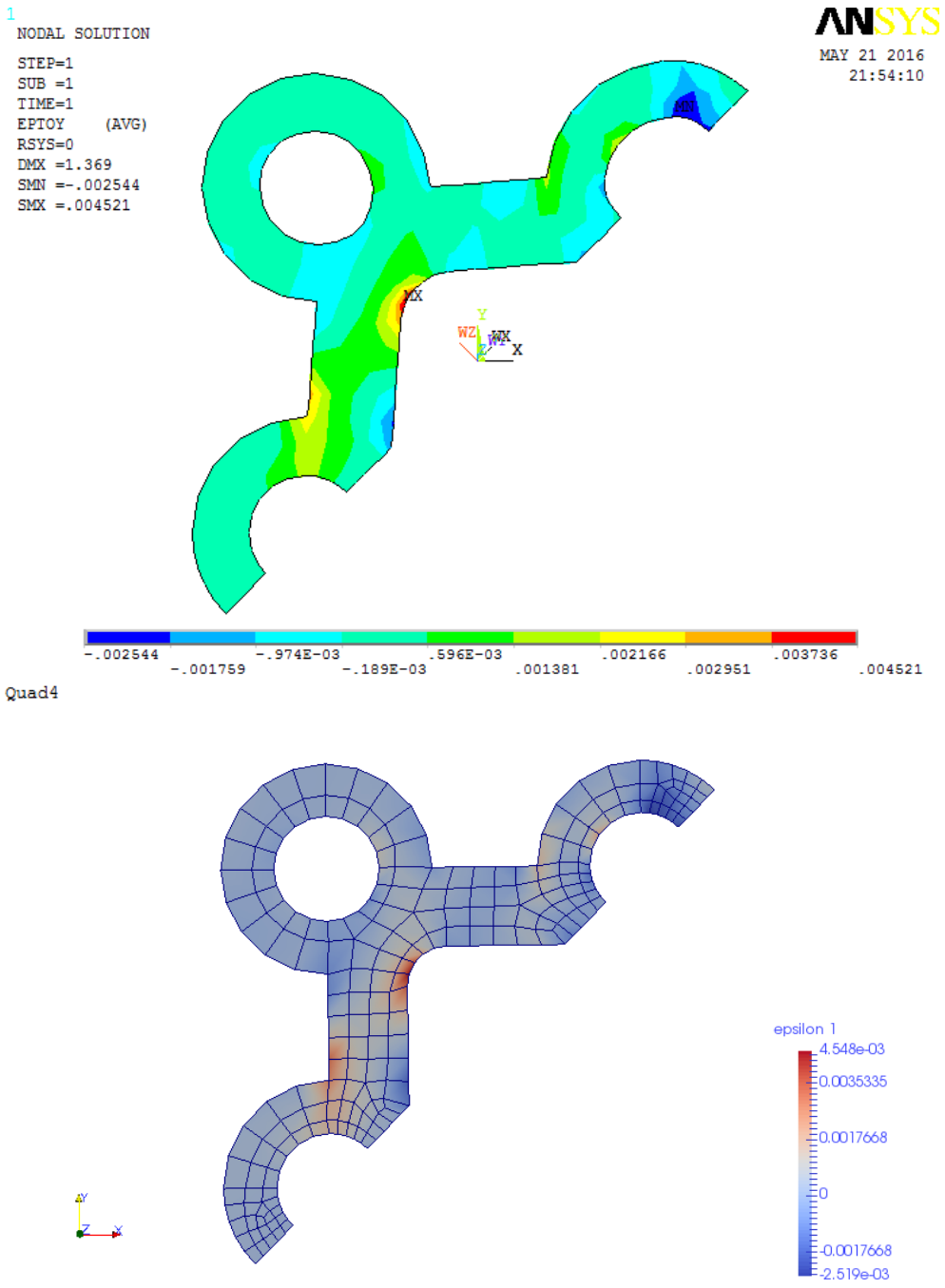


Figure 4.16: Mesh with Quad4. upper: contour plot of ϵ_{yy} from ANSYS; lower: contour plot of ϵ_{yy} calculated in AMfe, demonstrate in ParaView

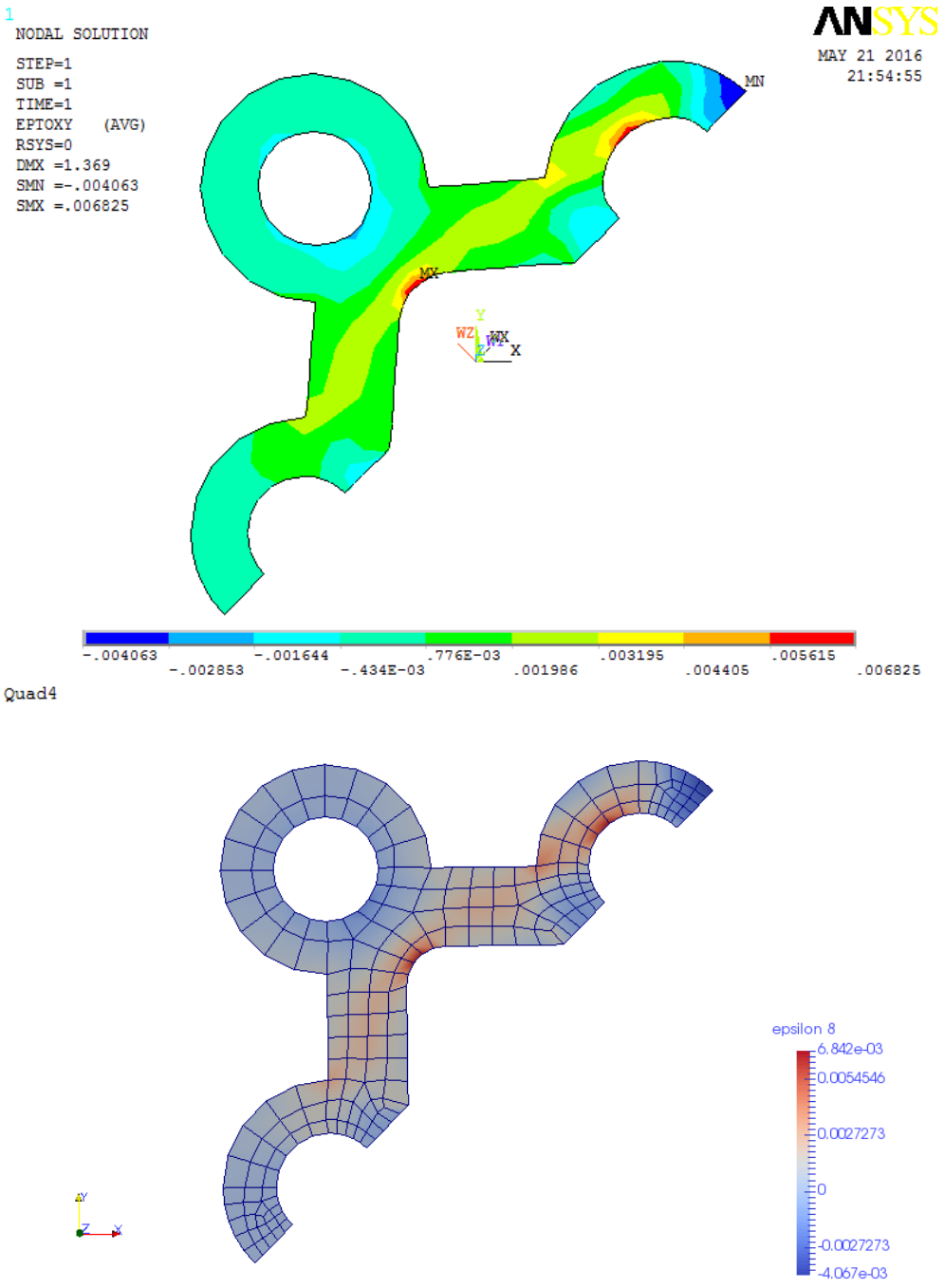


Figure 4.17: Mesh with Quad4. upper: contour plot of ϵ_{xy} from ANSYS; lower: contour plot of ϵ_{xy} calculated in AMfe, demonstrate in ParaView

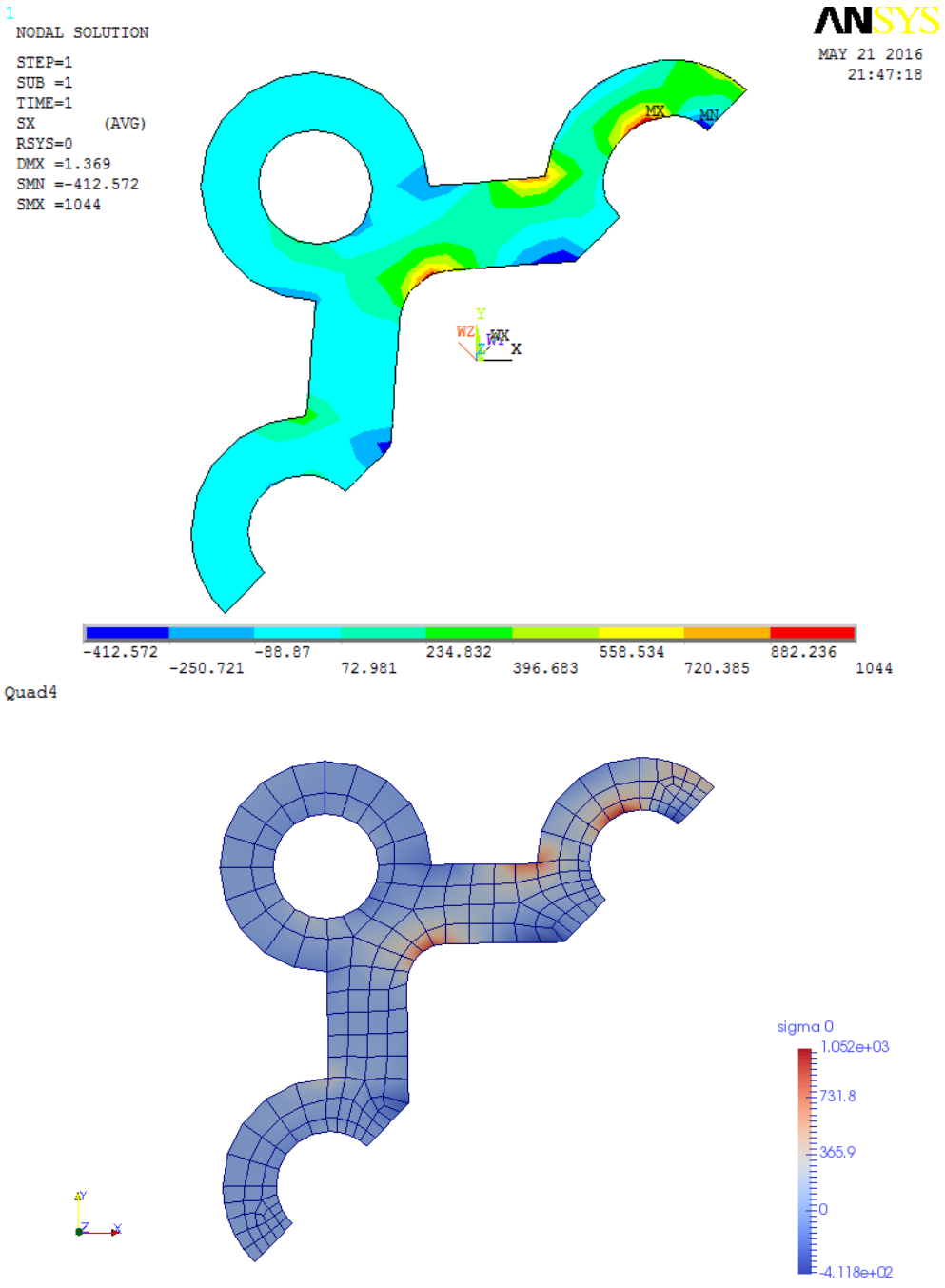


Figure 4.18: Mesh with Quad4. upper: contour plot of σ_{xx} from ANSYS; lower: contour plot of σ_{xx} calculated in AMfe, demonstrate in ParaView

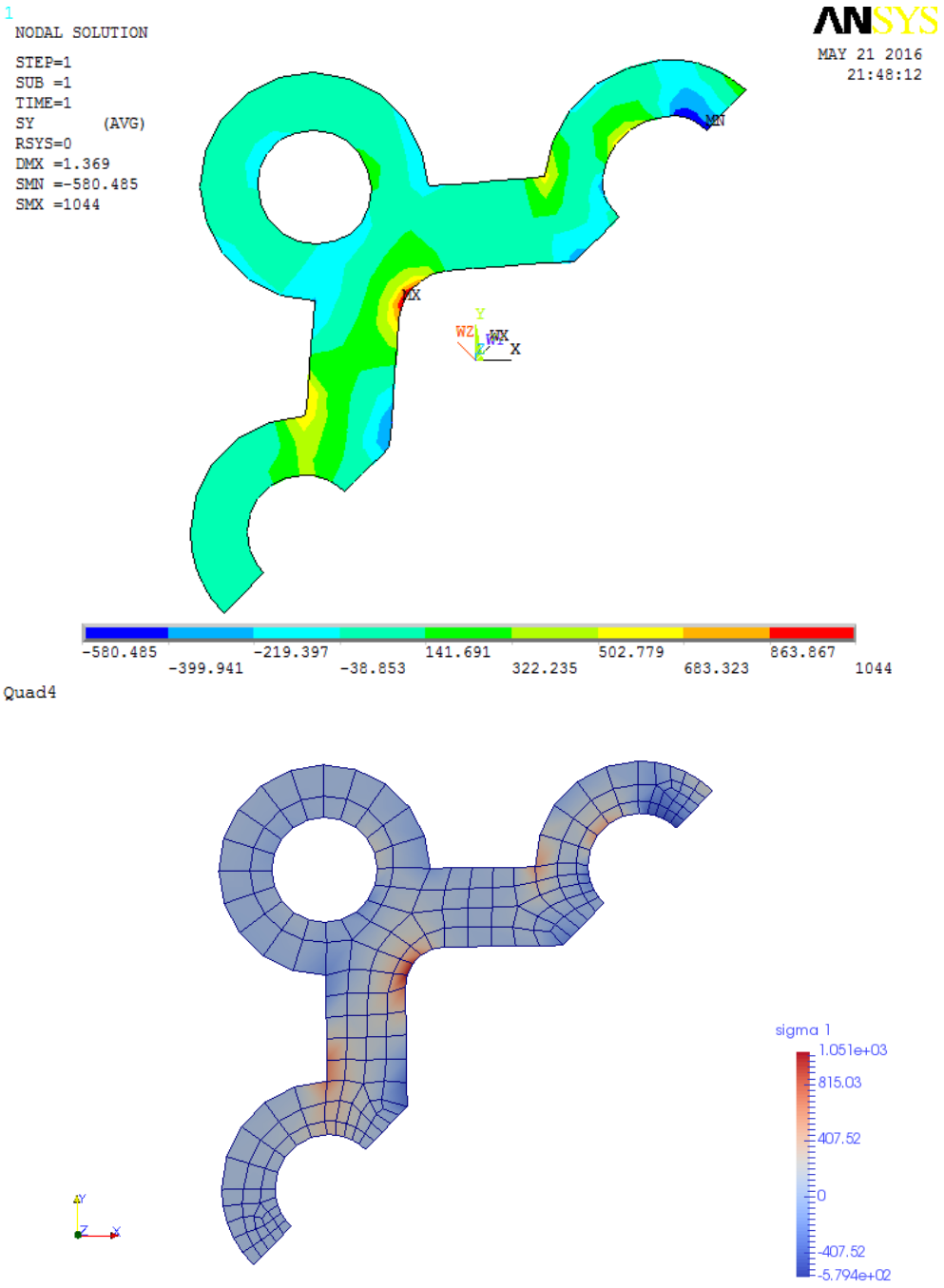


Figure 4.19: Mesh with Quad4. upper: contour plot of σ_{yy} from ANSYS; lower: contour plot of σ_{yy} calculated in AMfe, demonstrate in ParaView

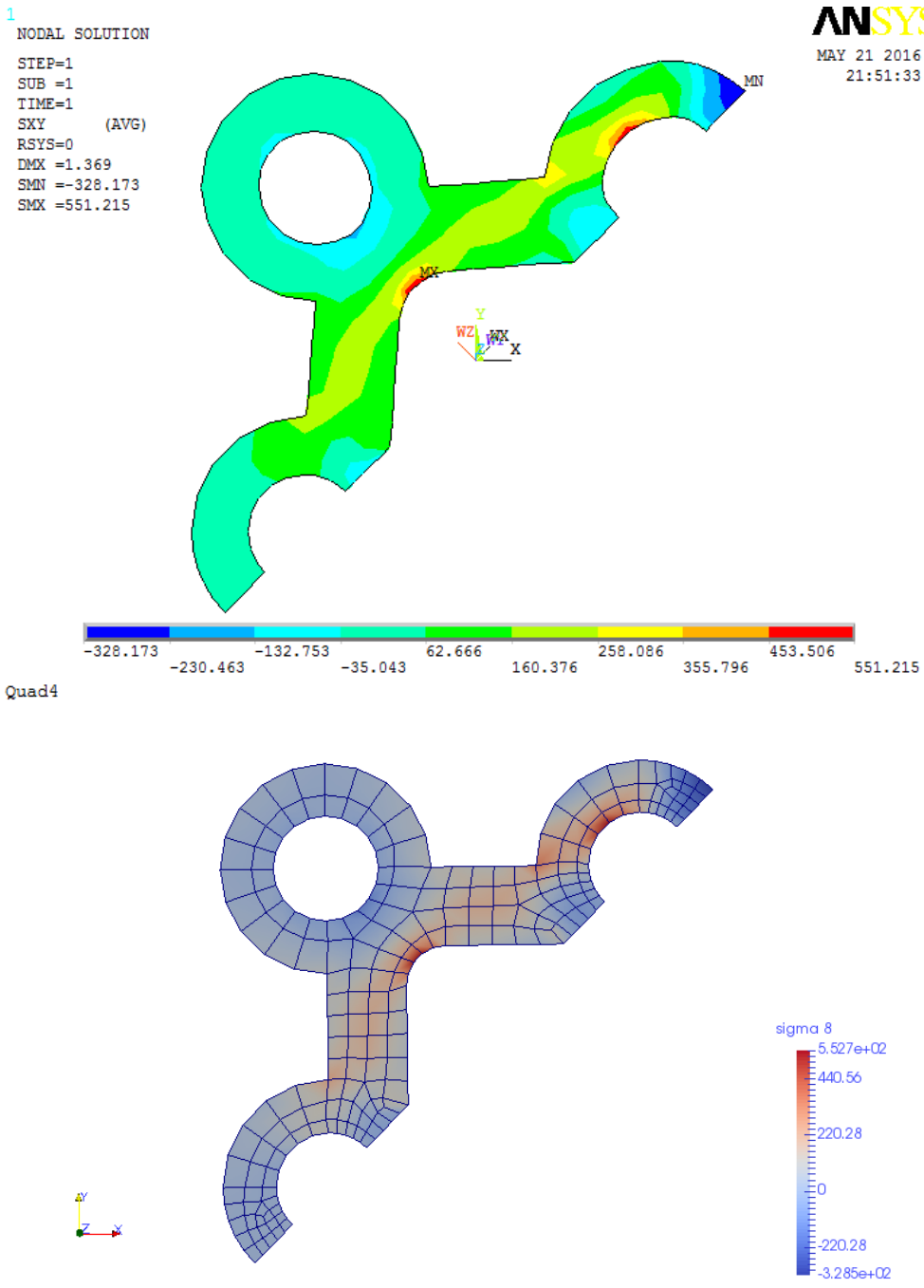


Figure 4.20: Mesh with Quad4. upper: contour plot of σ_{xy} from ANSYS; lower: contour plot of σ_{xy} calculated in AMfe, demonstrate in ParaView

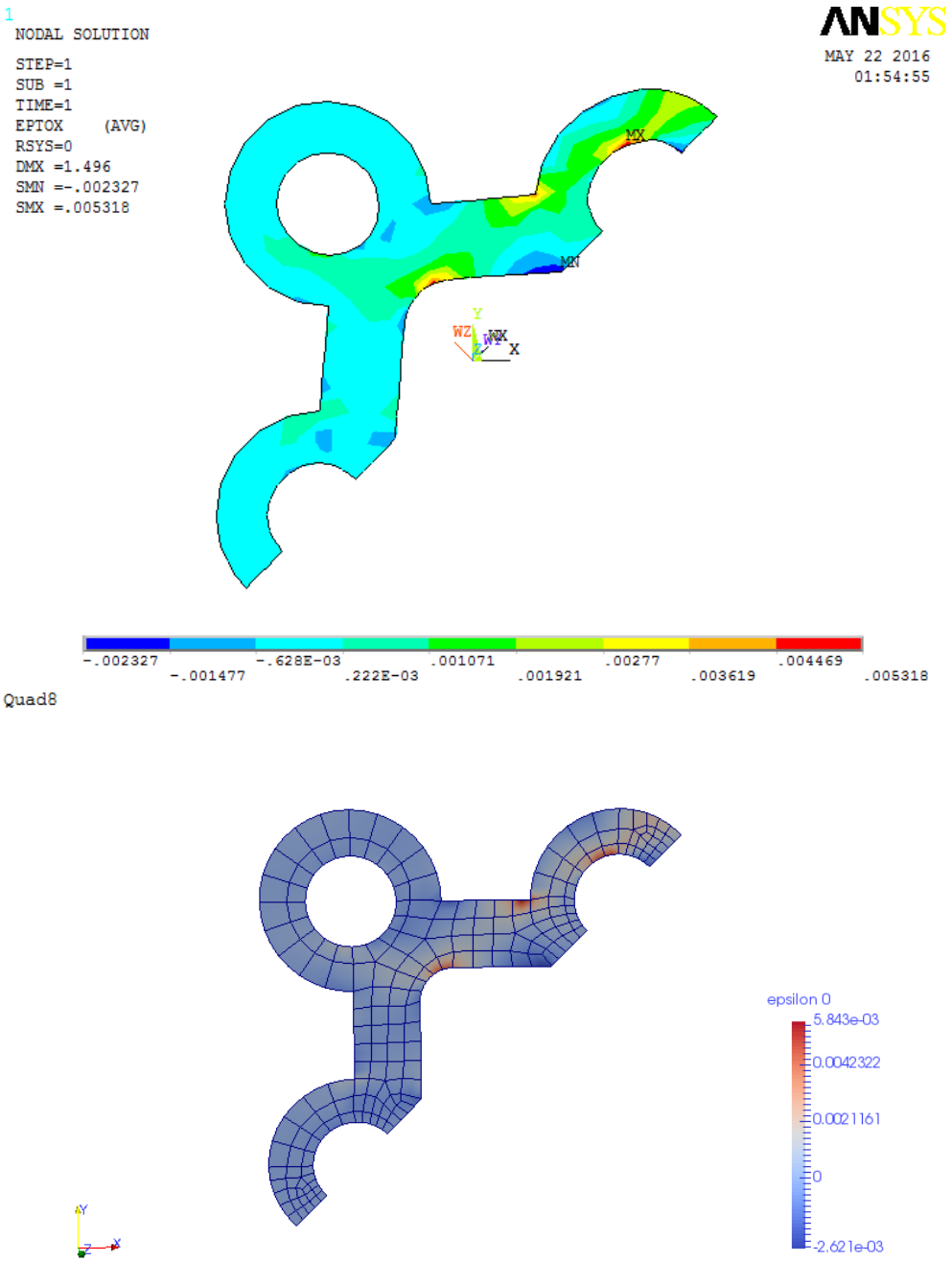


Figure 4.21: Mesh with Quad8. upper: contour plot of ϵ_{xx} from ANSYS; lower: contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

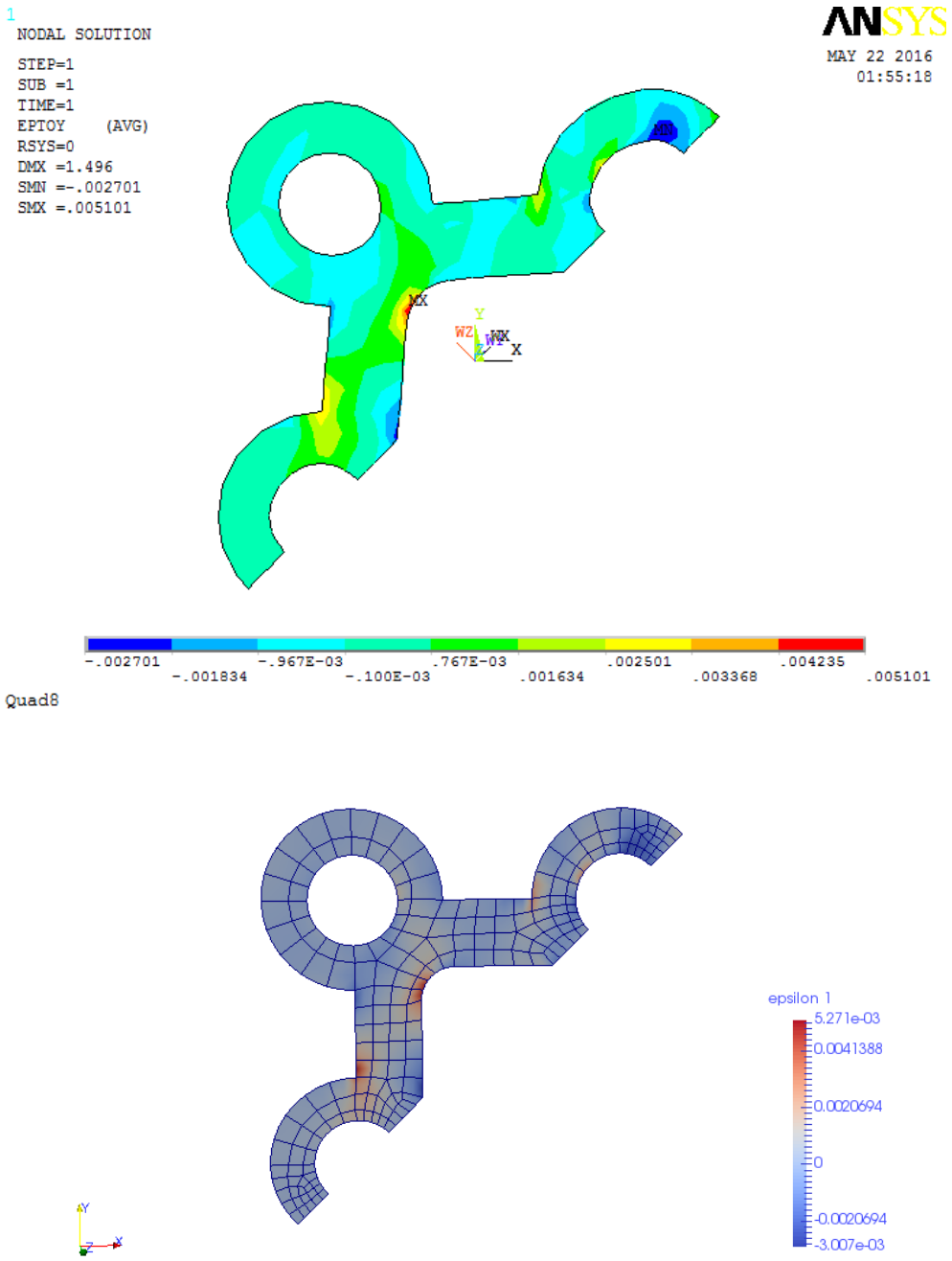


Figure 4.22: Mesh with Quad8. upper: contour plot of ϵ_{yy} from ANSYS; lower: contour plot of ϵ_{yy} calculated in AMfe, demonstrate in ParaView

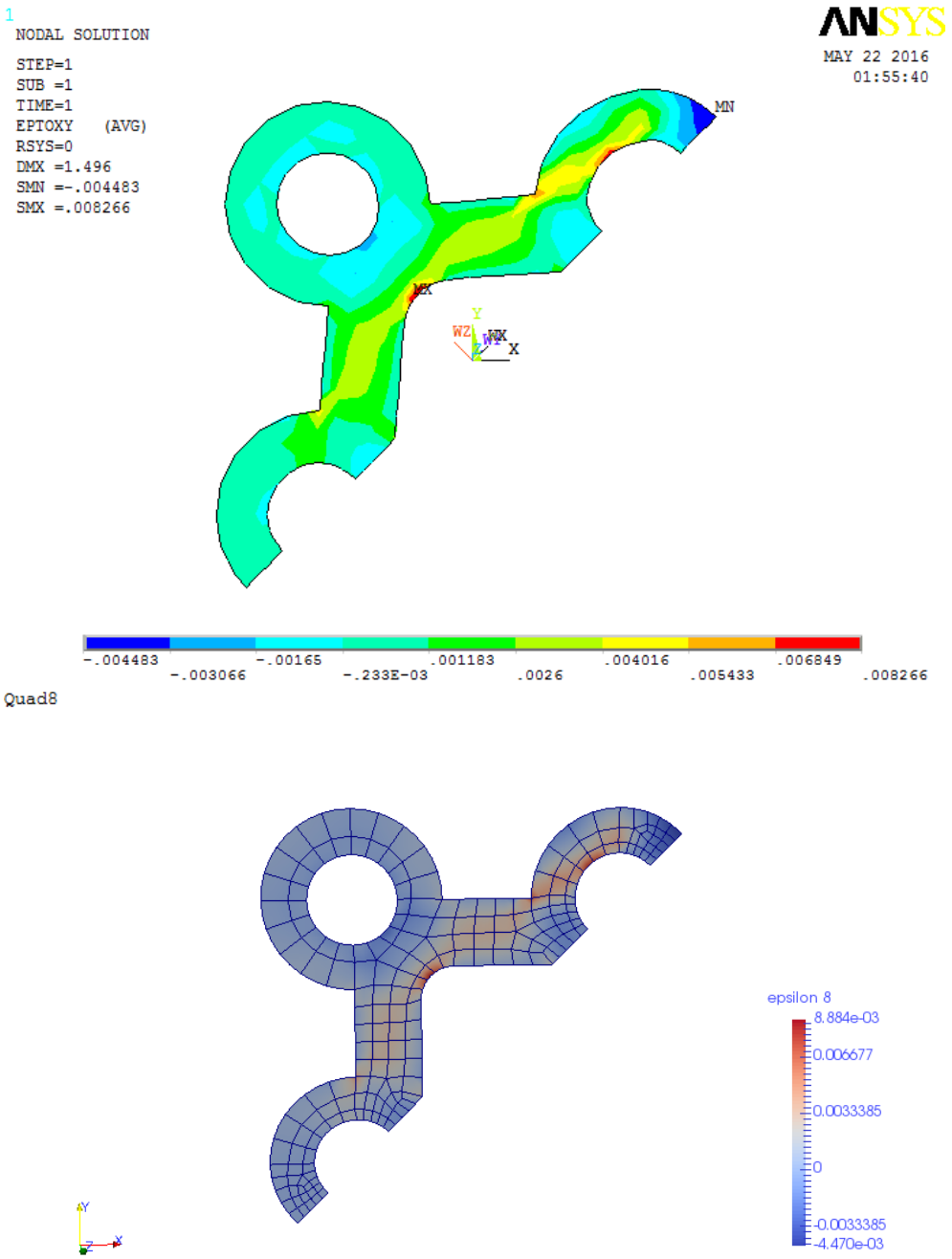


Figure 4.23: Mesh with Quad8. upper: contour plot of ϵ_{xy} from ANSYS; lower: contour plot of ϵ_{xy} calculated in AMfe, demonstrate in ParaView

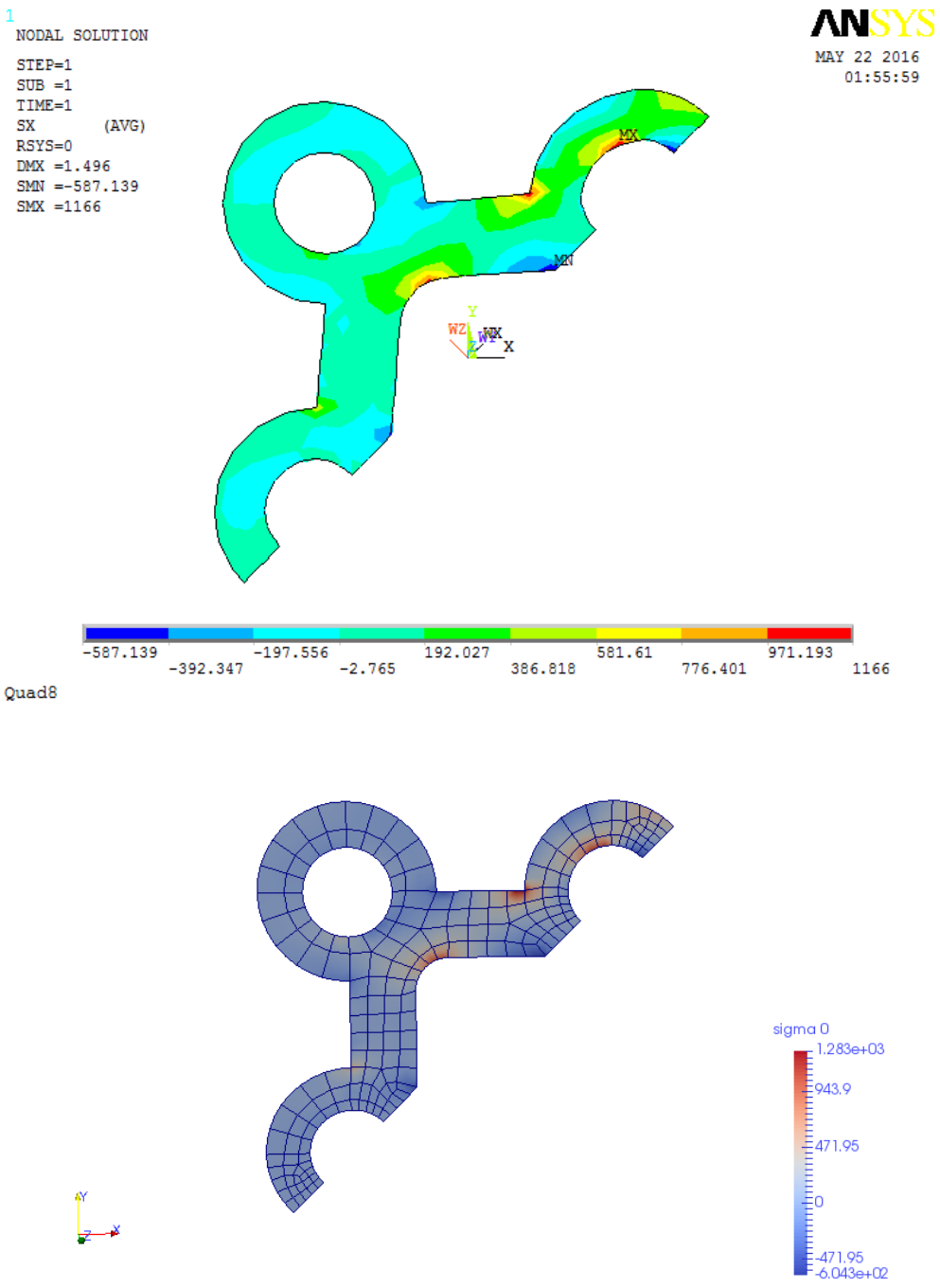


Figure 4.24: Mesh with Quad8. upper: contour plot of σ_{xx} from ANSYS; lower: contour plot of σ_{xx} calculated in AMfe, demonstrate in ParaView

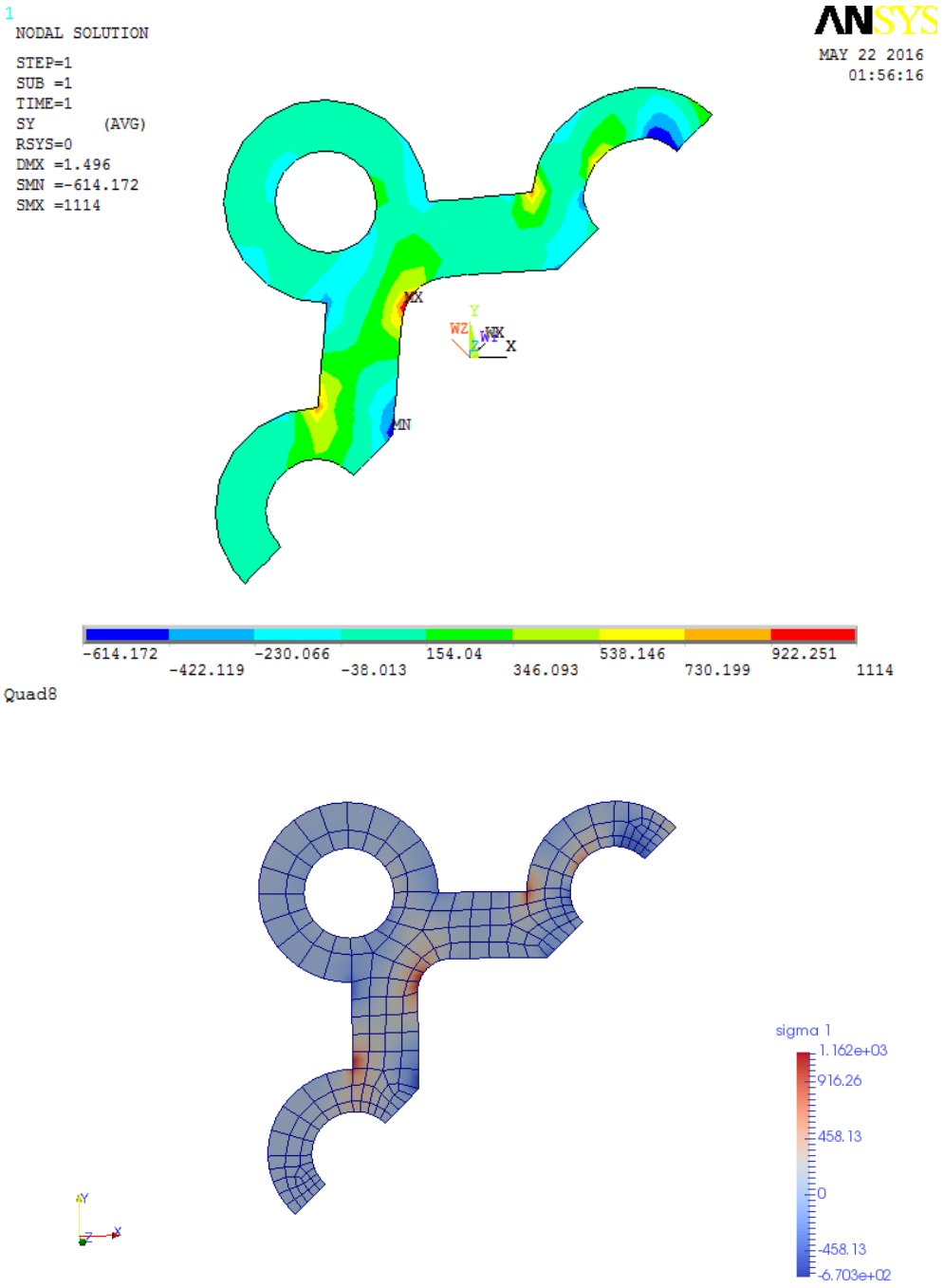


Figure 4.25: Mesh with Quad8. upper: contour plot of σ_{yy} from ANSYS; lower: contour plot of σ_{yy} calculated in AMfe, demonstrate in ParaView

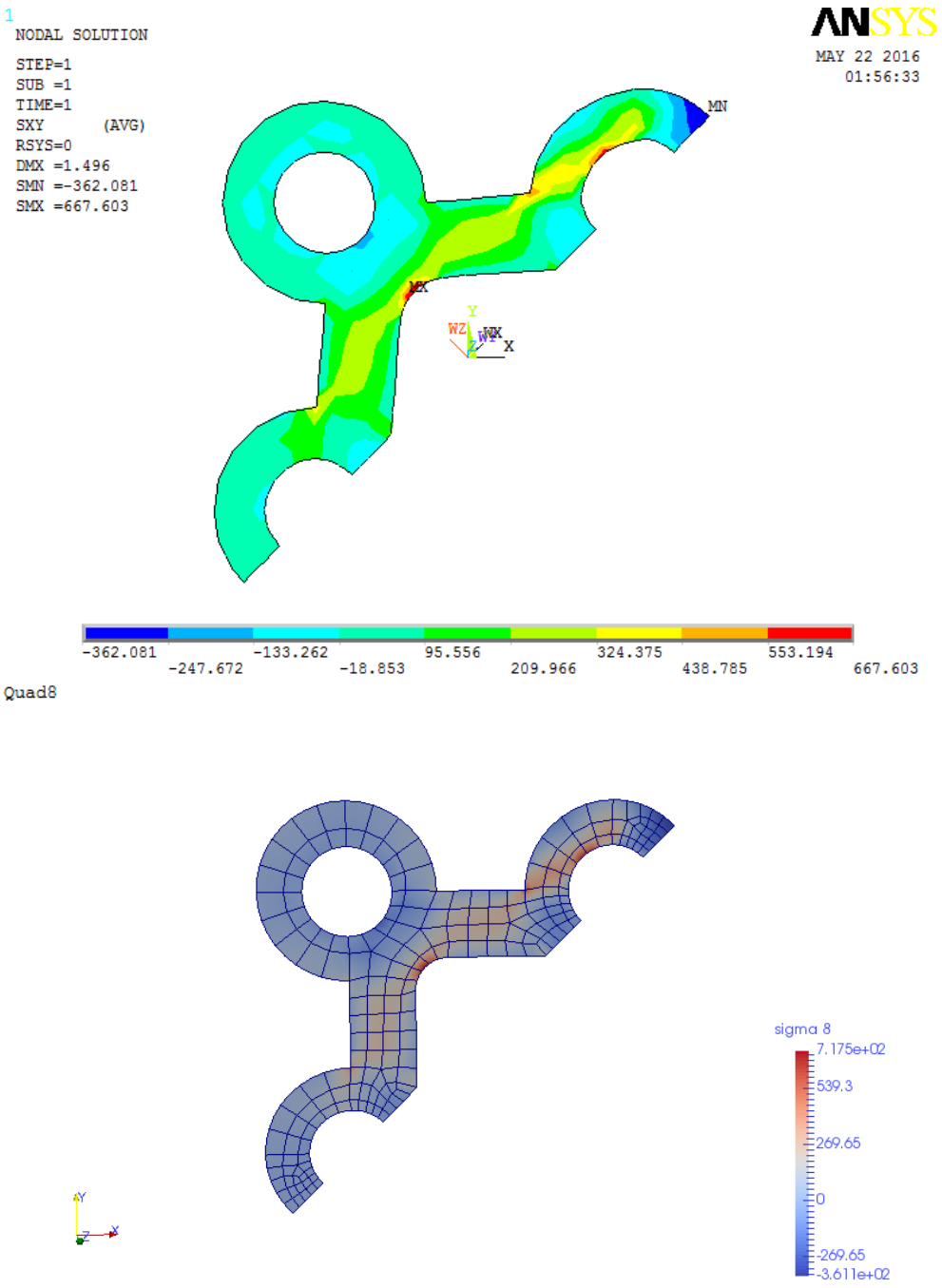


Figure 4.26: Mesh with Quad8. upper: contour plot of σ_{xy} from ANSYS; lower: contour plot of σ_{xy} calculated in AMfe, demonstrate in ParaView

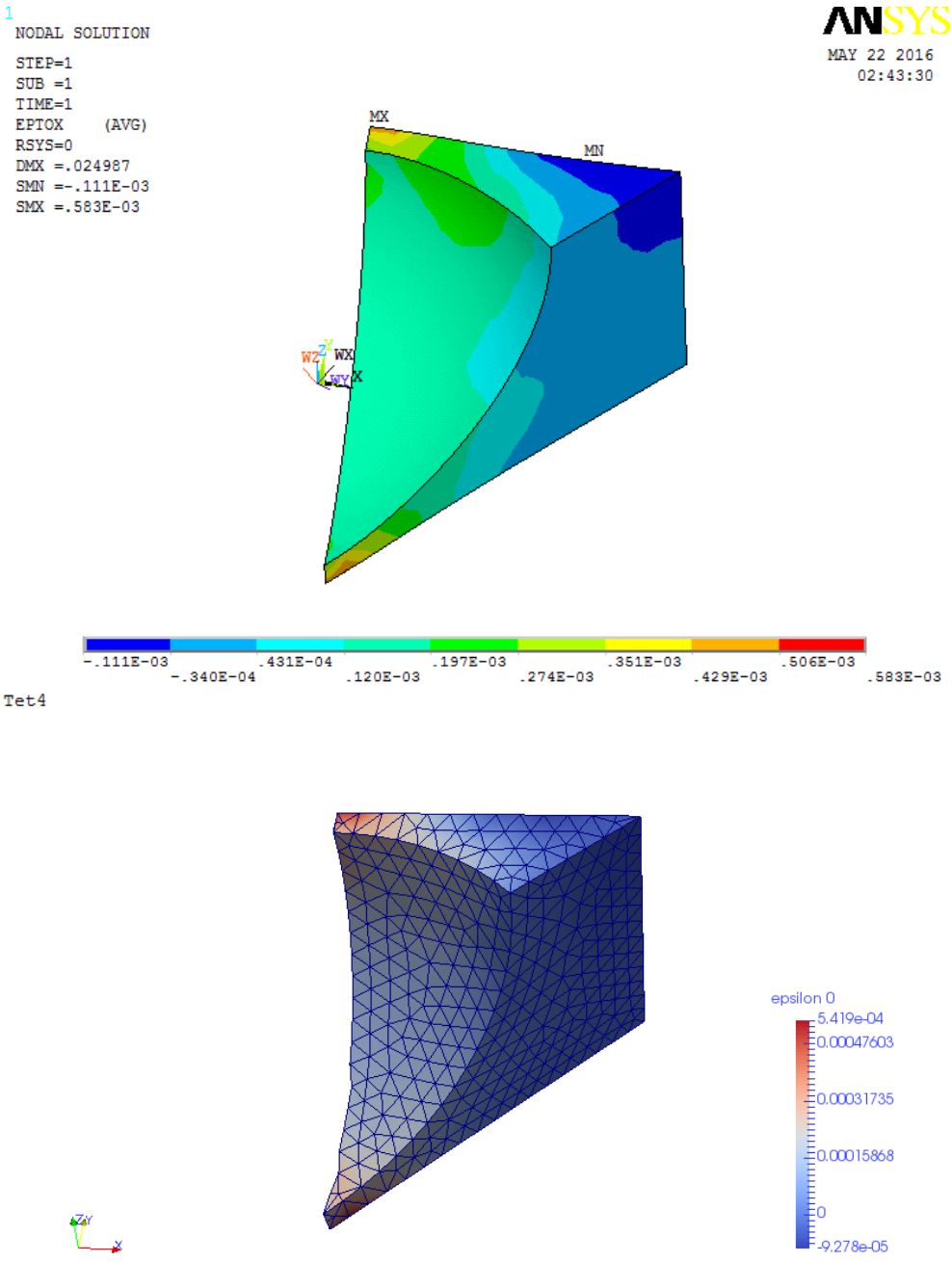


Figure 4.27: Mesh with Tet4. upper: contour plot of ϵ_{xx} from ANSYS; lower: contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

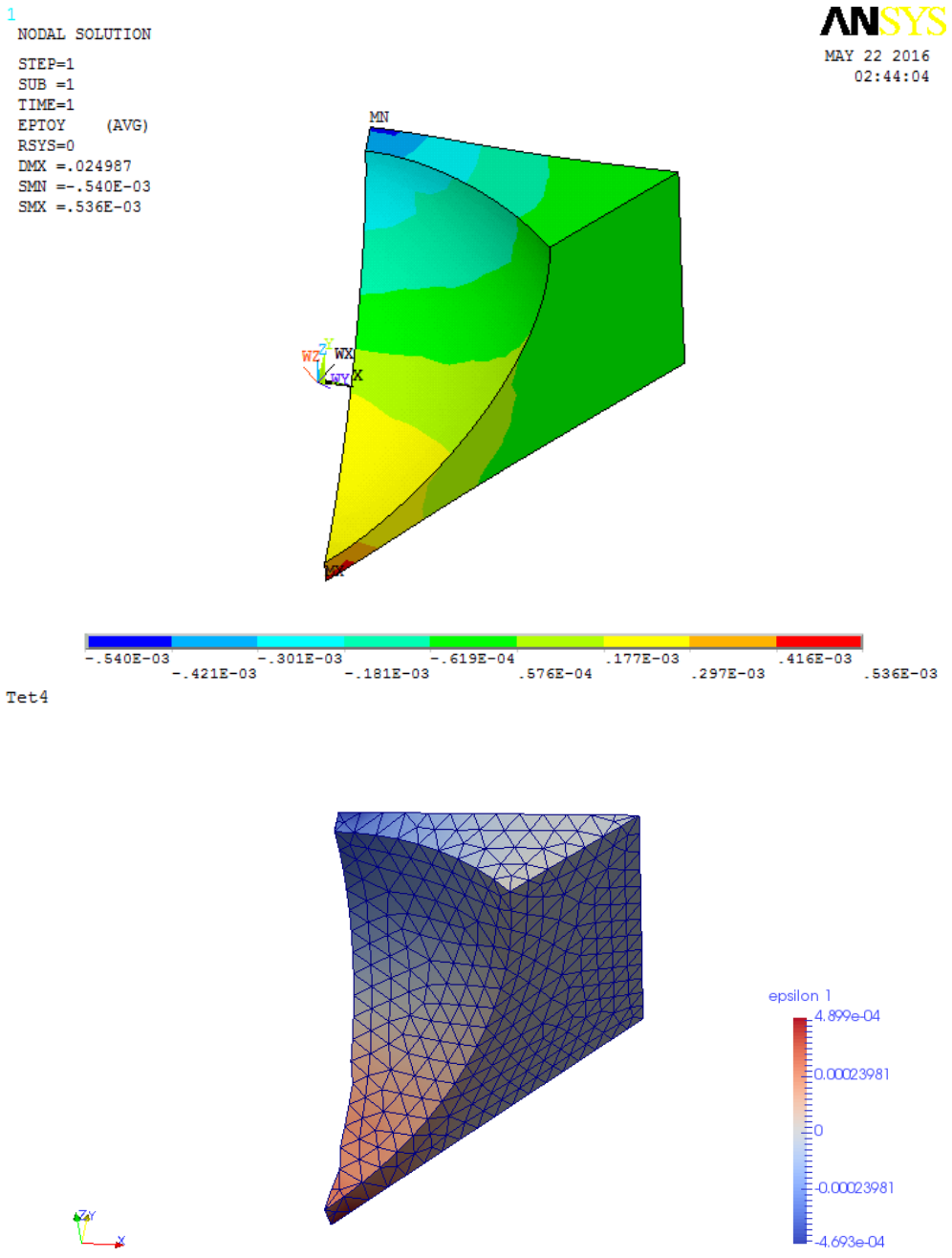


Figure 4.28: Mesh with Tet4. upper: contour plot of ϵ_{yy} from ANSYS; lower: contour plot of ϵ_{yy} calculated in AMfe, demonstrate in ParaView

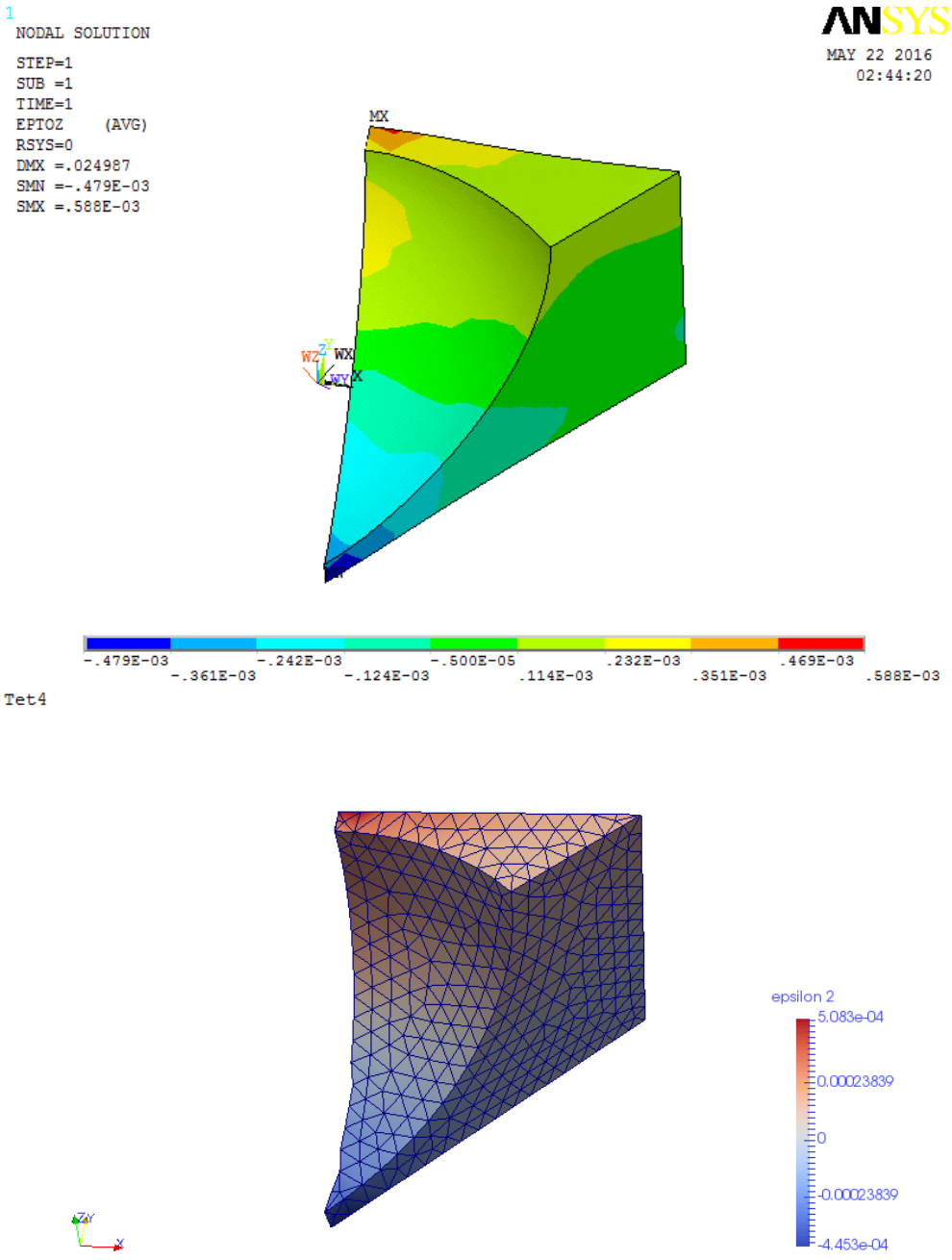


Figure 4.29: Mesh with Tet4. upper: contour plot of ϵ_{zz} from ANSYS; lower: contour plot of ϵ_{zz} calculated in AMfe, demonstrate in ParaView

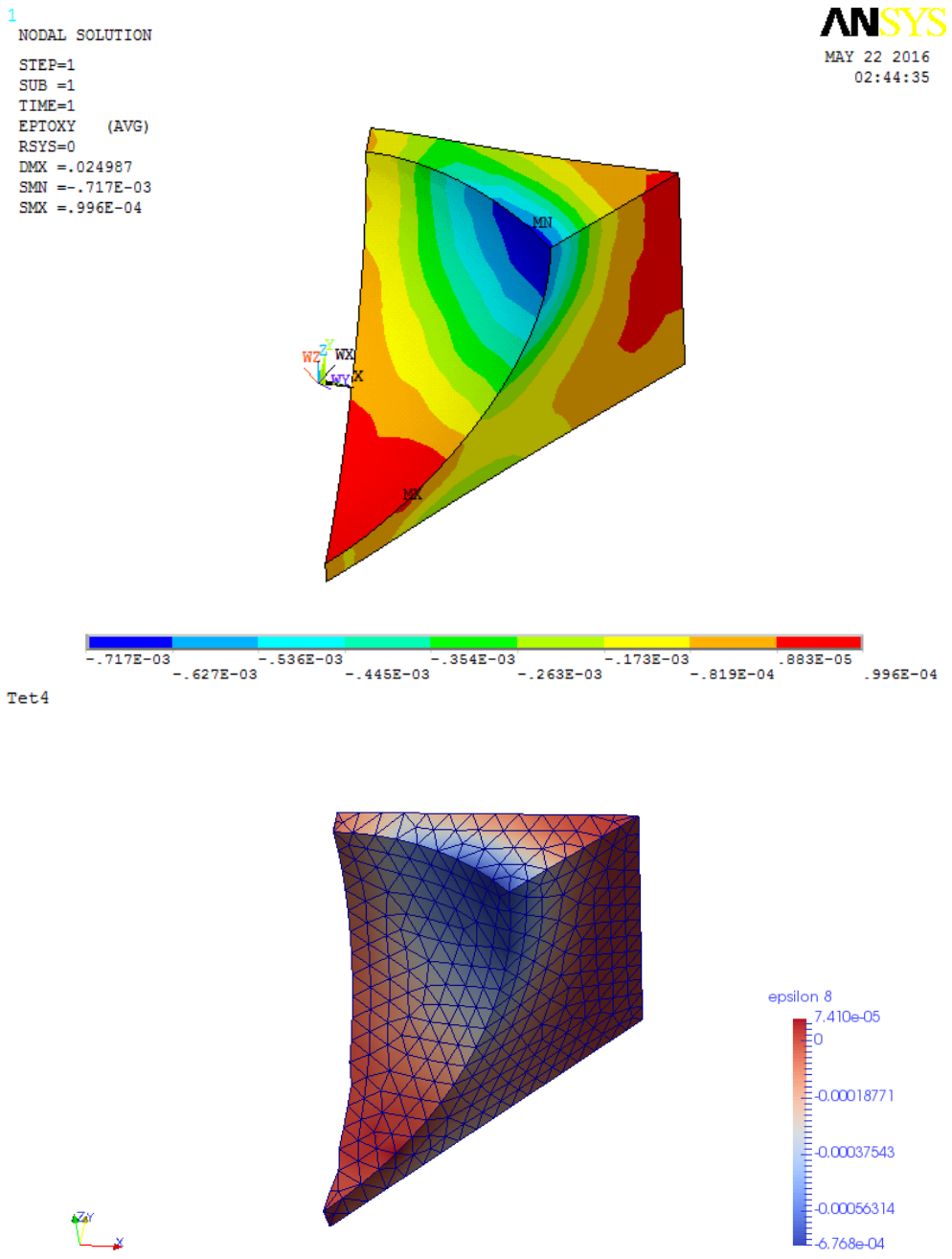


Figure 4.30: Mesh with Tet4. upper: contour plot of ϵ_{xy} from ANSYS; lower: contour plot of ϵ_{xy} calculated in AMfe, demonstrate in ParaView

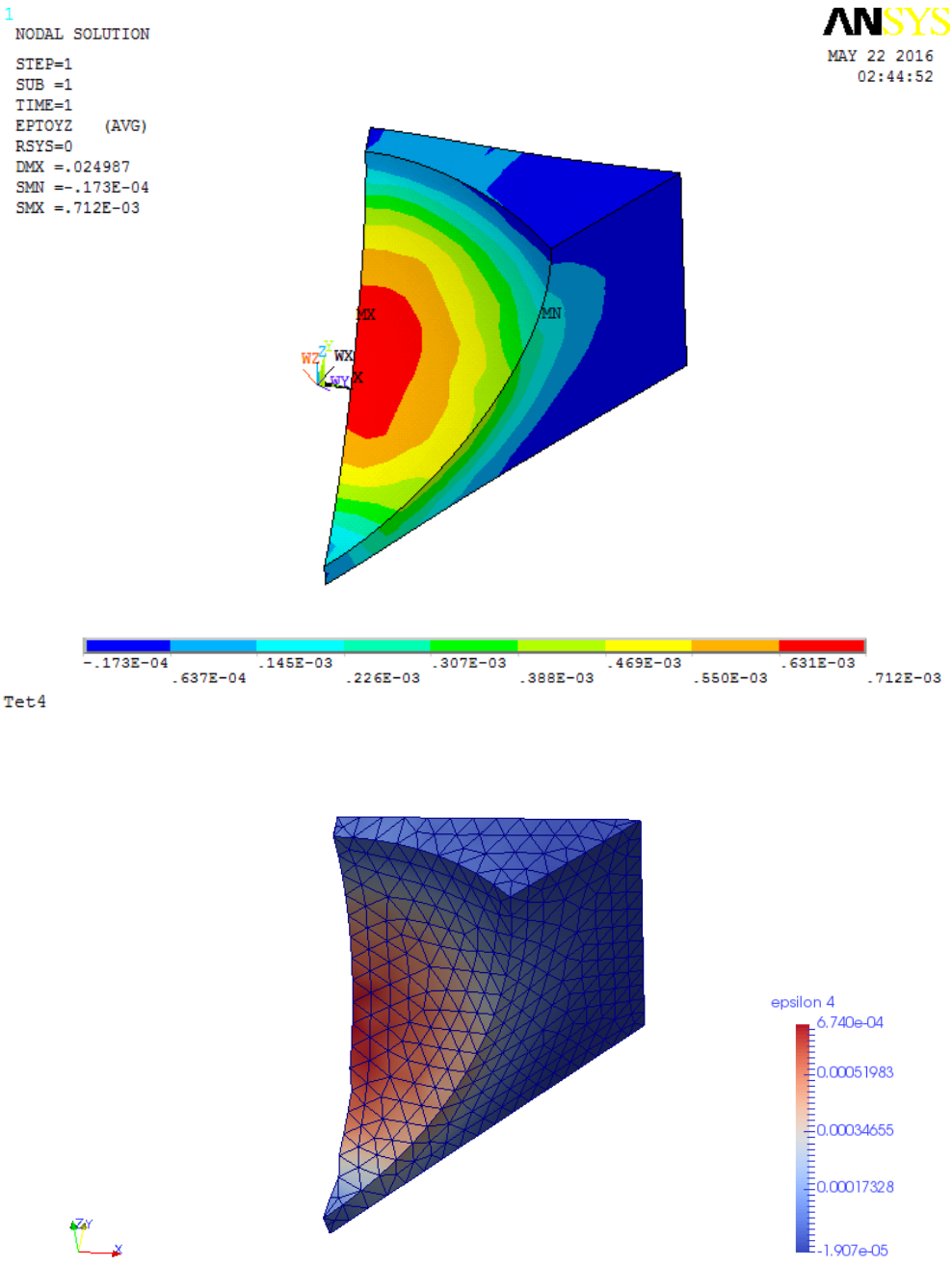


Figure 4.31: Mesh with Tet4. upper: contour plot of ϵ_{yz} from ANSYS; lower: contour plot of ϵ_{yz} calculated in AMfe, demonstrate in ParaView

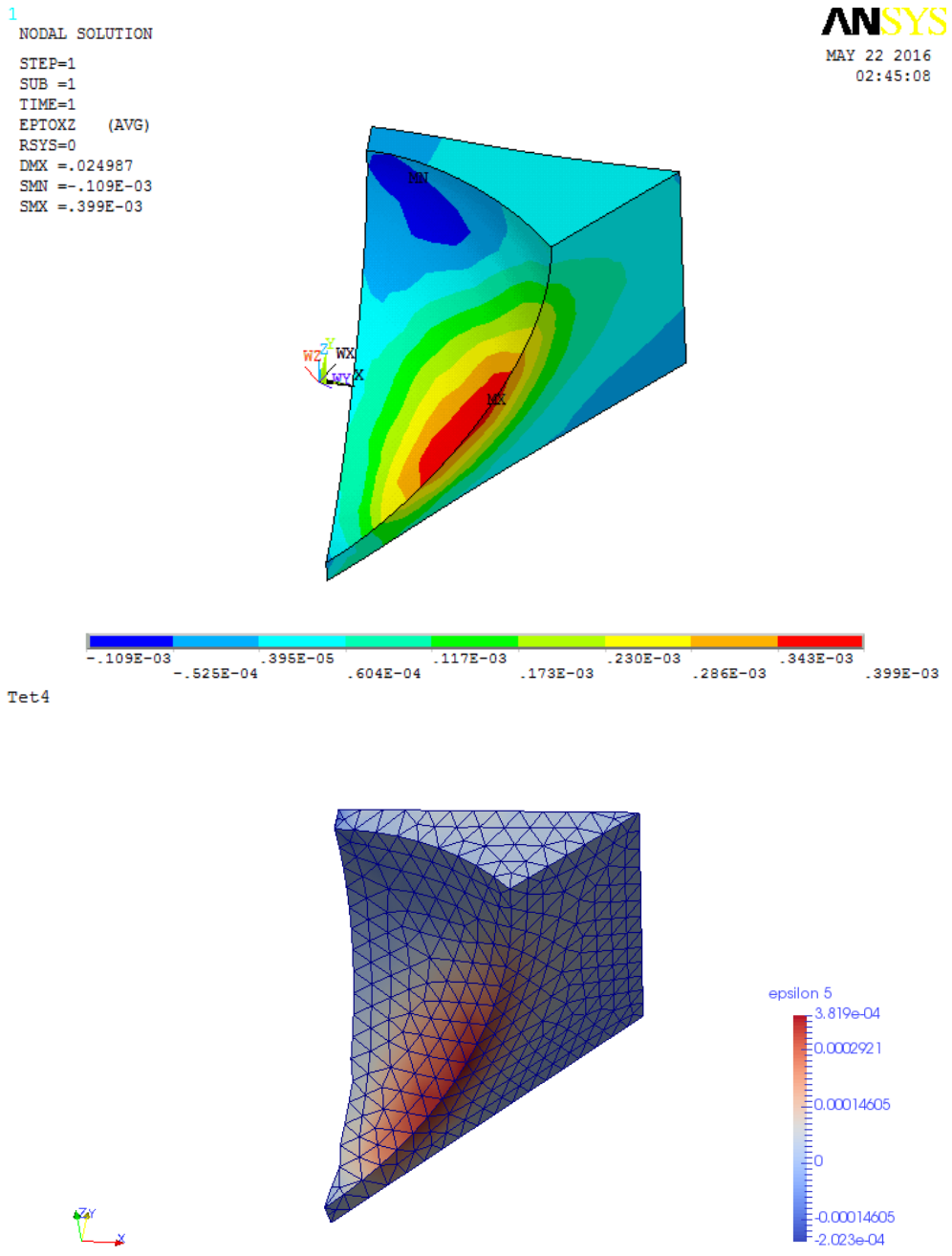


Figure 4.32: Mesh with Tet4. upper: contour plot of ϵ_{xz} from ANSYS; lower: contour plot of ϵ_{xz} calculated in AMfe, demonstrate in ParaView

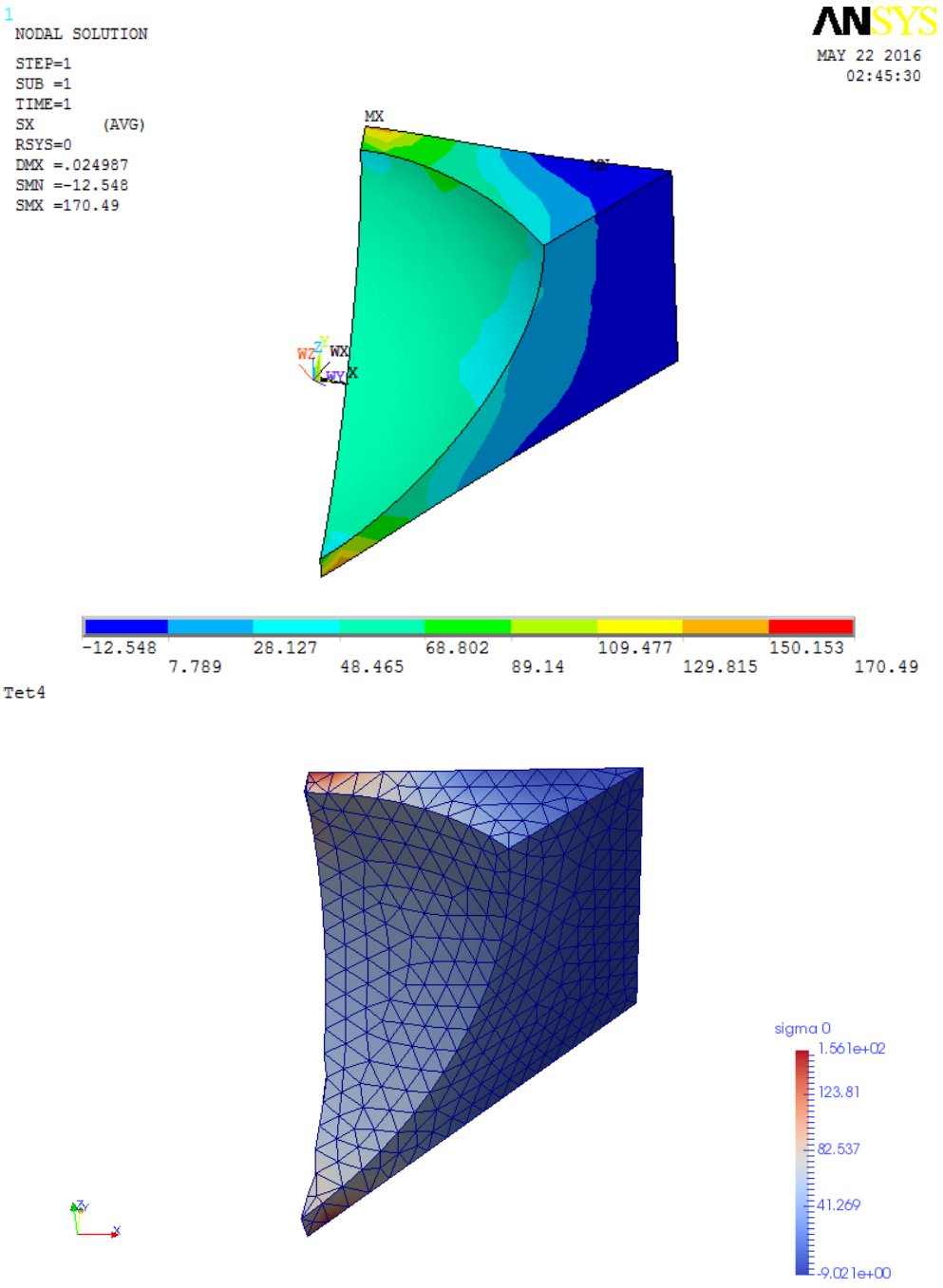


Figure 4.33: Mesh with Tet4. upper: contour plot of σ_{xx} from ANSYS; lower: contour plot of σ_{xx} calculated in AMfe, demonstrate in ParaView

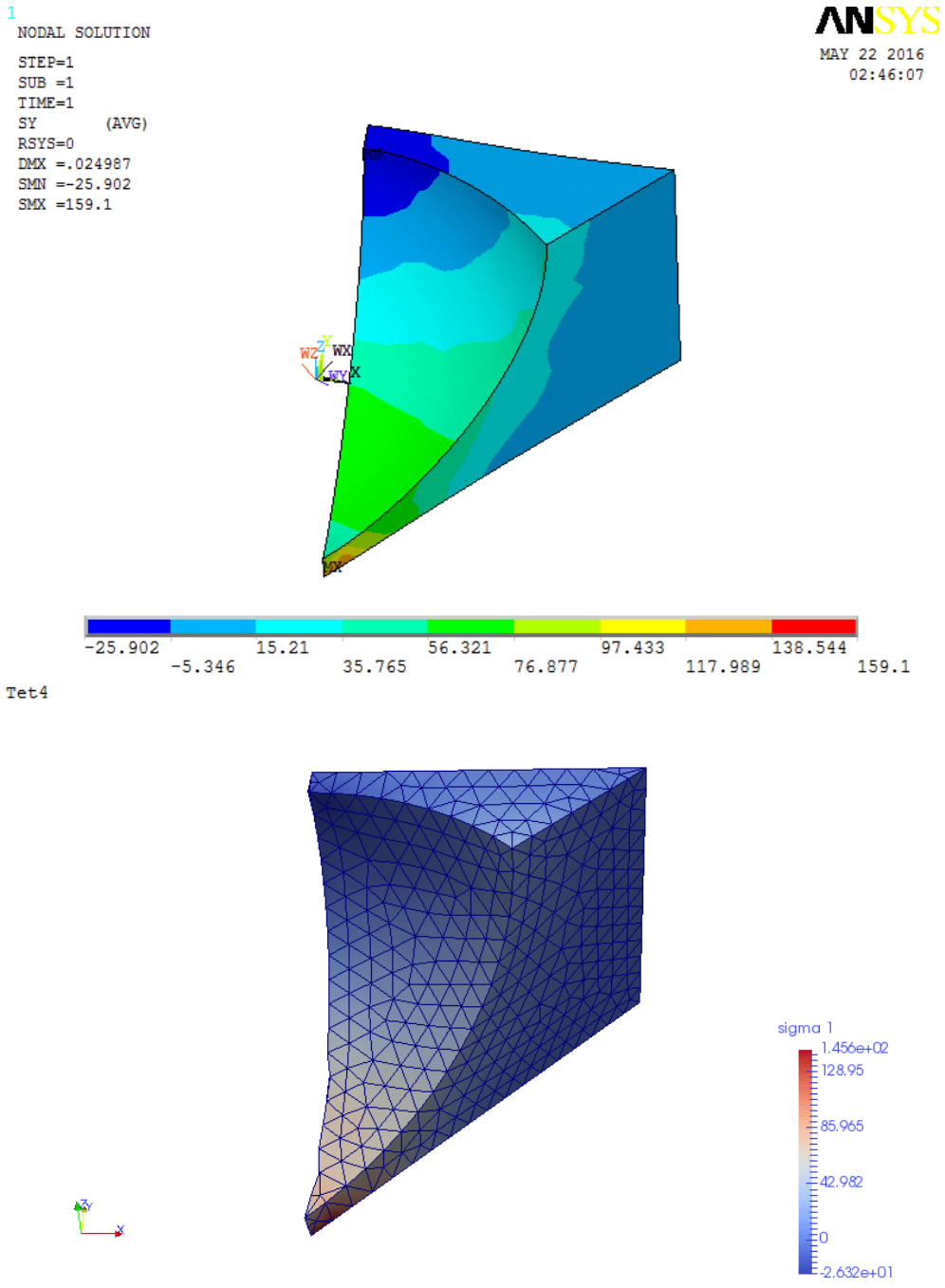


Figure 4.34: Mesh with Tet4. upper: contour plot of σ_{yy} from ANSYS; lower: contour plot of σ_{yy} calculated in AMfe, demonstrate in ParaView

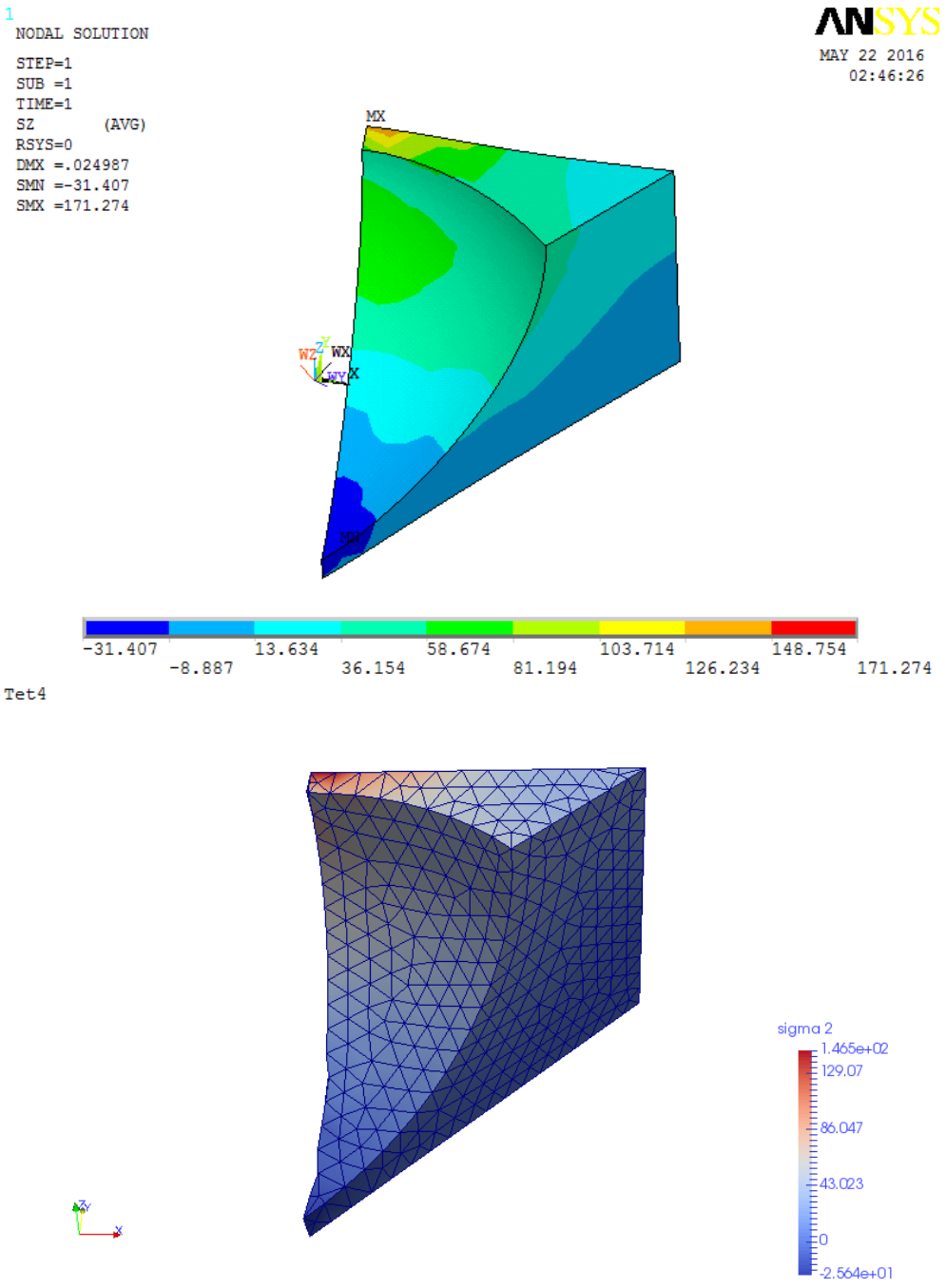


Figure 4.35: Mesh with Tet4. upper: contour plot of σ_{zz} from ANSYS; lower: contour plot of σ_{zz} calculated in AMfe, demonstrate in ParaView

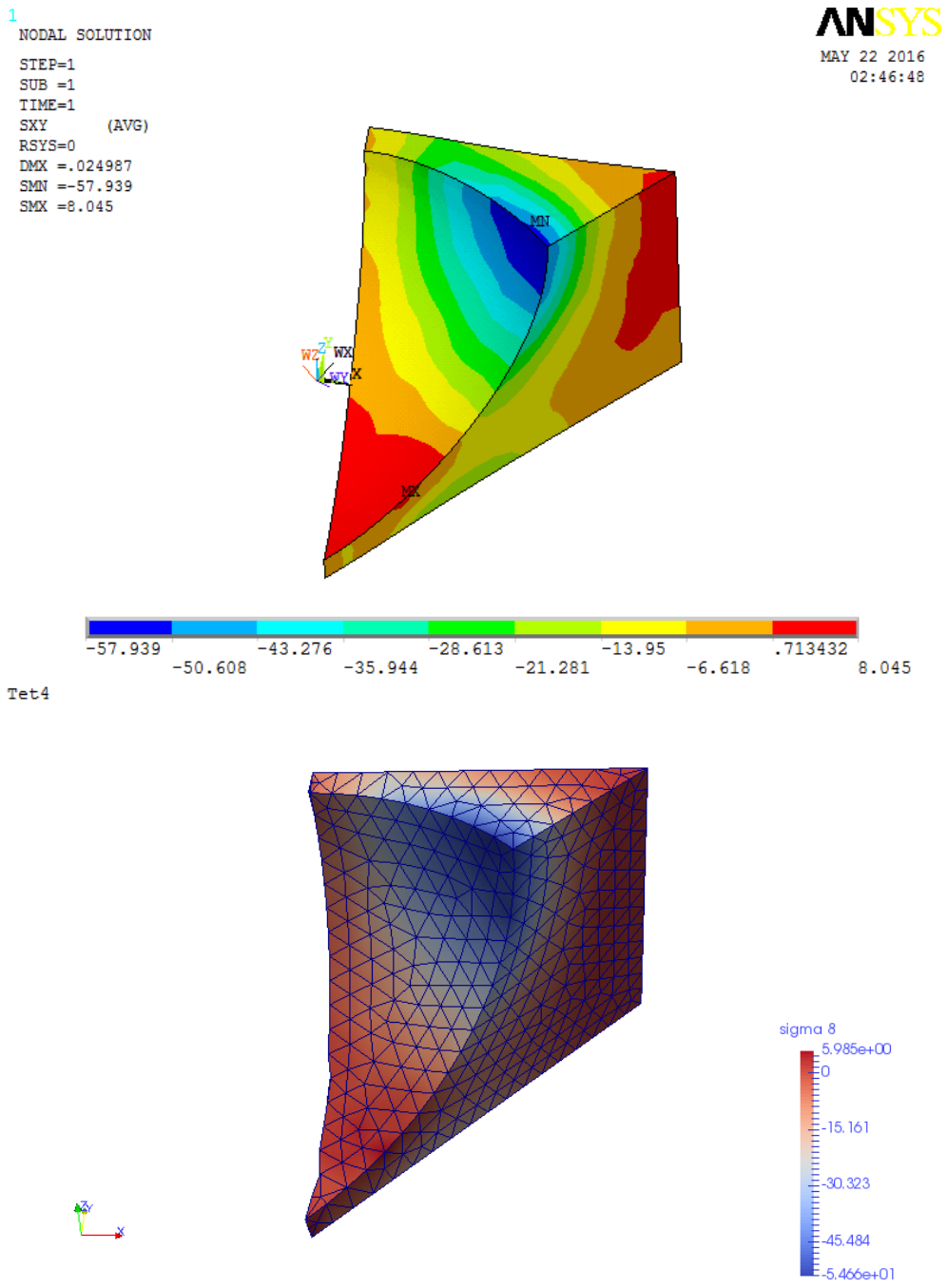


Figure 4.36: Mesh with Tet4. upper: contour plot of σ_{xy} from ANSYS; lower: contour plot of σ_{xy} calculated in AMfe, demonstrate in ParaView

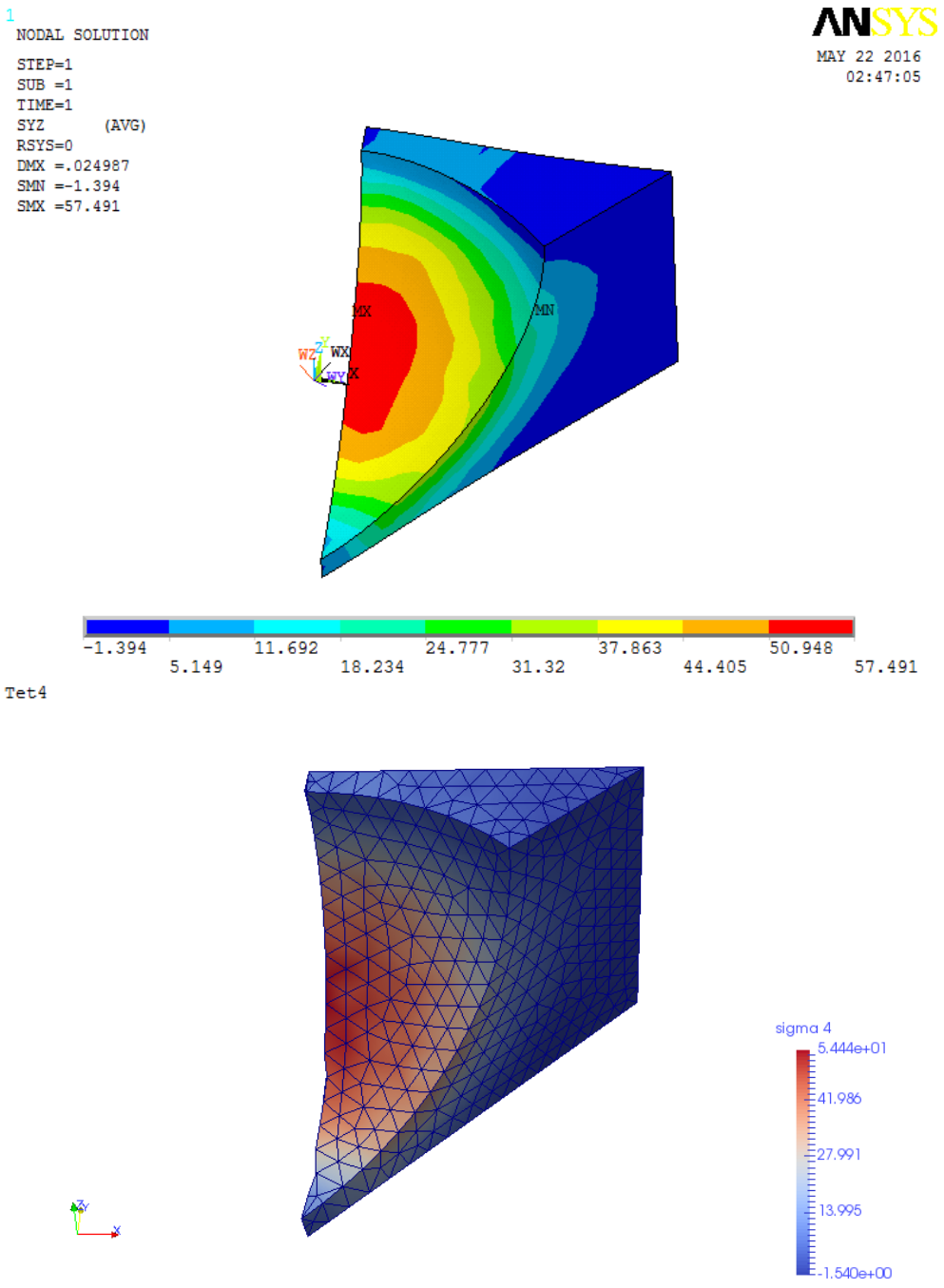


Figure 4.37: Mesh with Tet4. upper: contour plot of σ_{yz} from ANSYS; lower: contour plot of σ_{yz} calculated in AMfe, demonstrate in ParaView

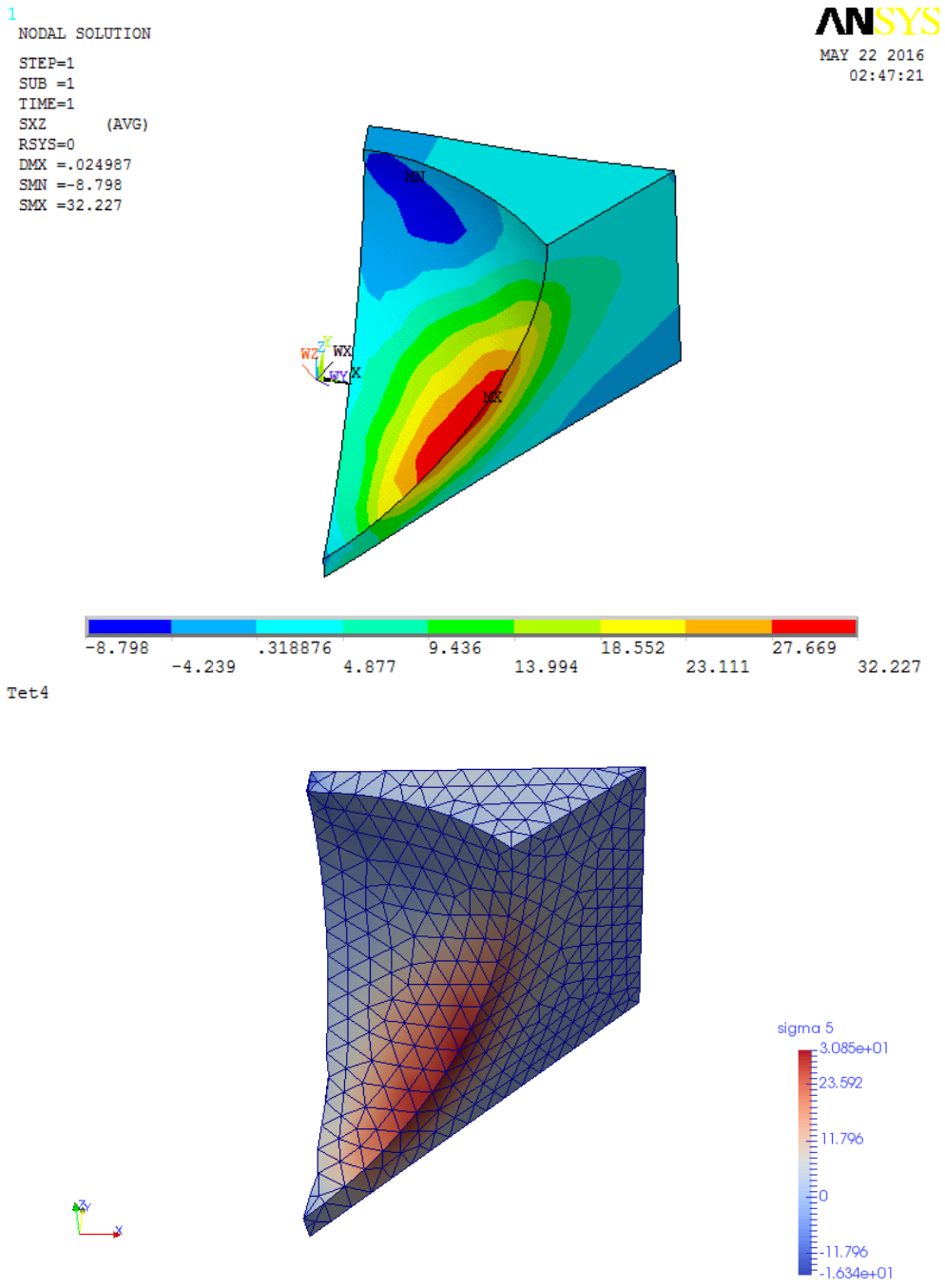


Figure 4.38: Mesh with Tet4. upper: contour plot of σ_{xz} from ANSYS; lower: contour plot of σ_{xz} calculated in AMfe, demonstrate in ParaView

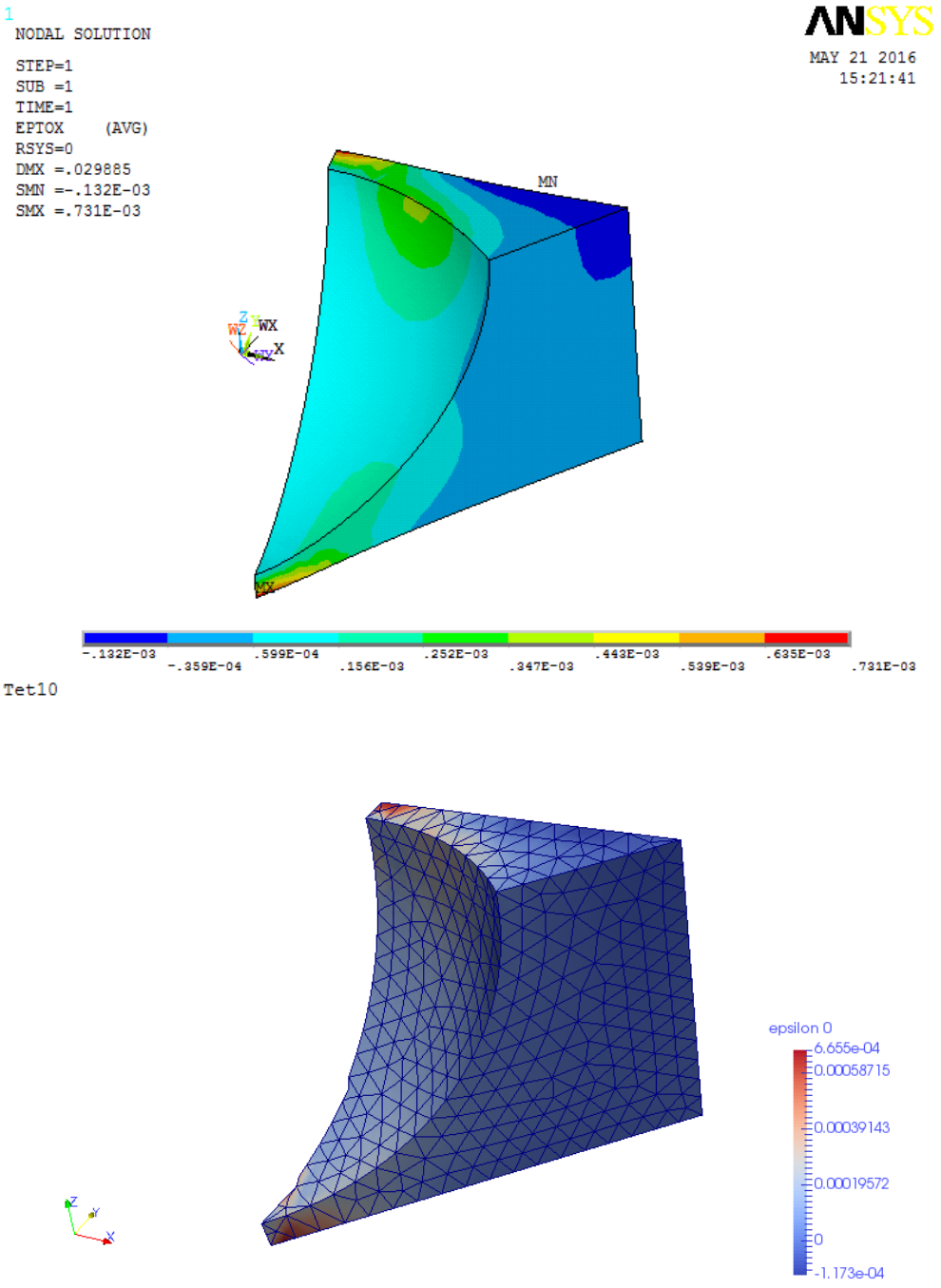


Figure 4.39: Mesh with Tet10. upper: contour plot of ϵ_{xx} from ANSYS; lower: contour plot of ϵ_{xx} calculated in AMfe, demonstrate in ParaView

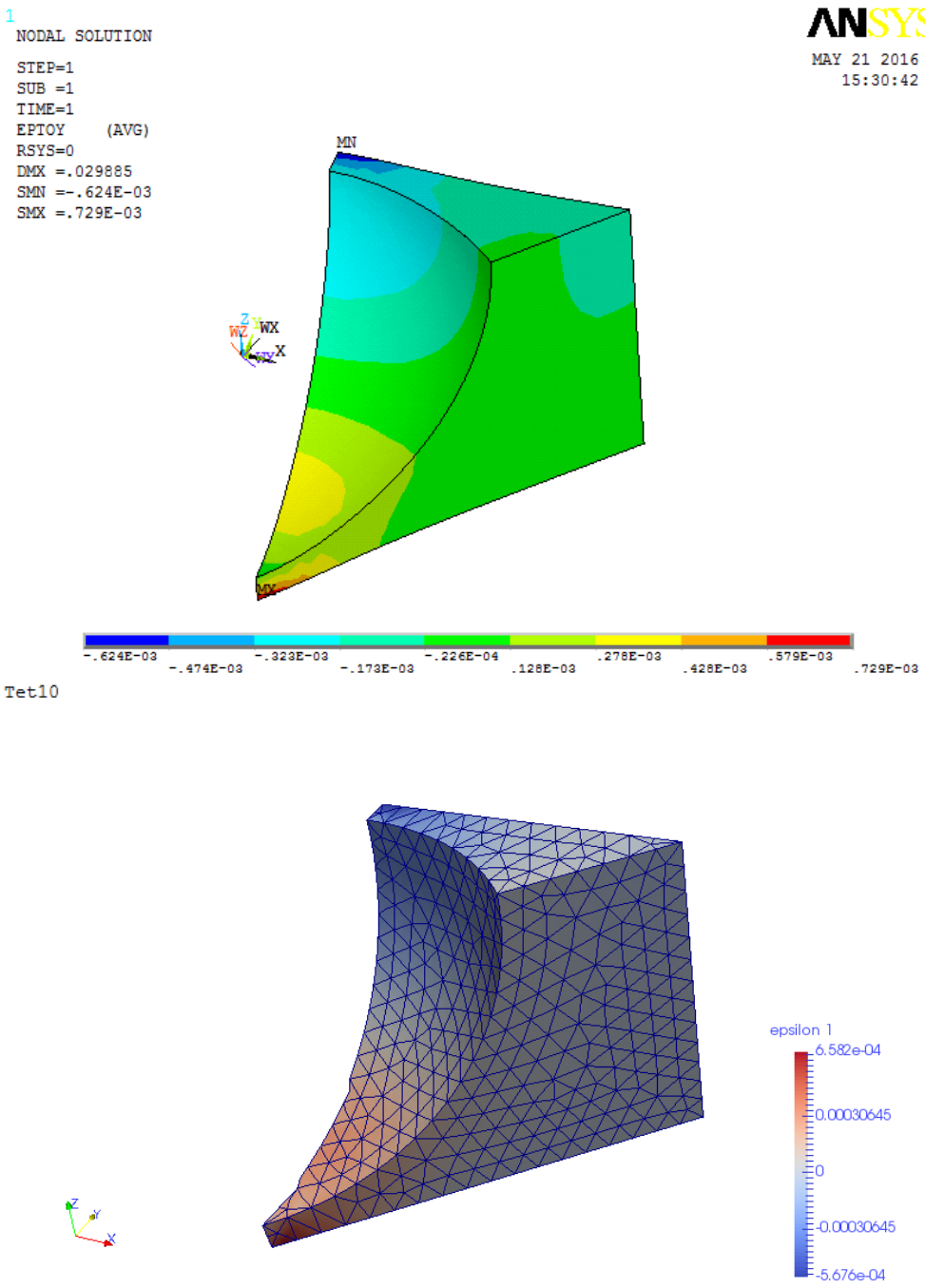


Figure 4.40: Mesh with Tet10. upper: contour plot of ϵ_{yy} from ANSYS; lower: contour plot of ϵ_{yy} calculated in AMfe, demonstrate in ParaView

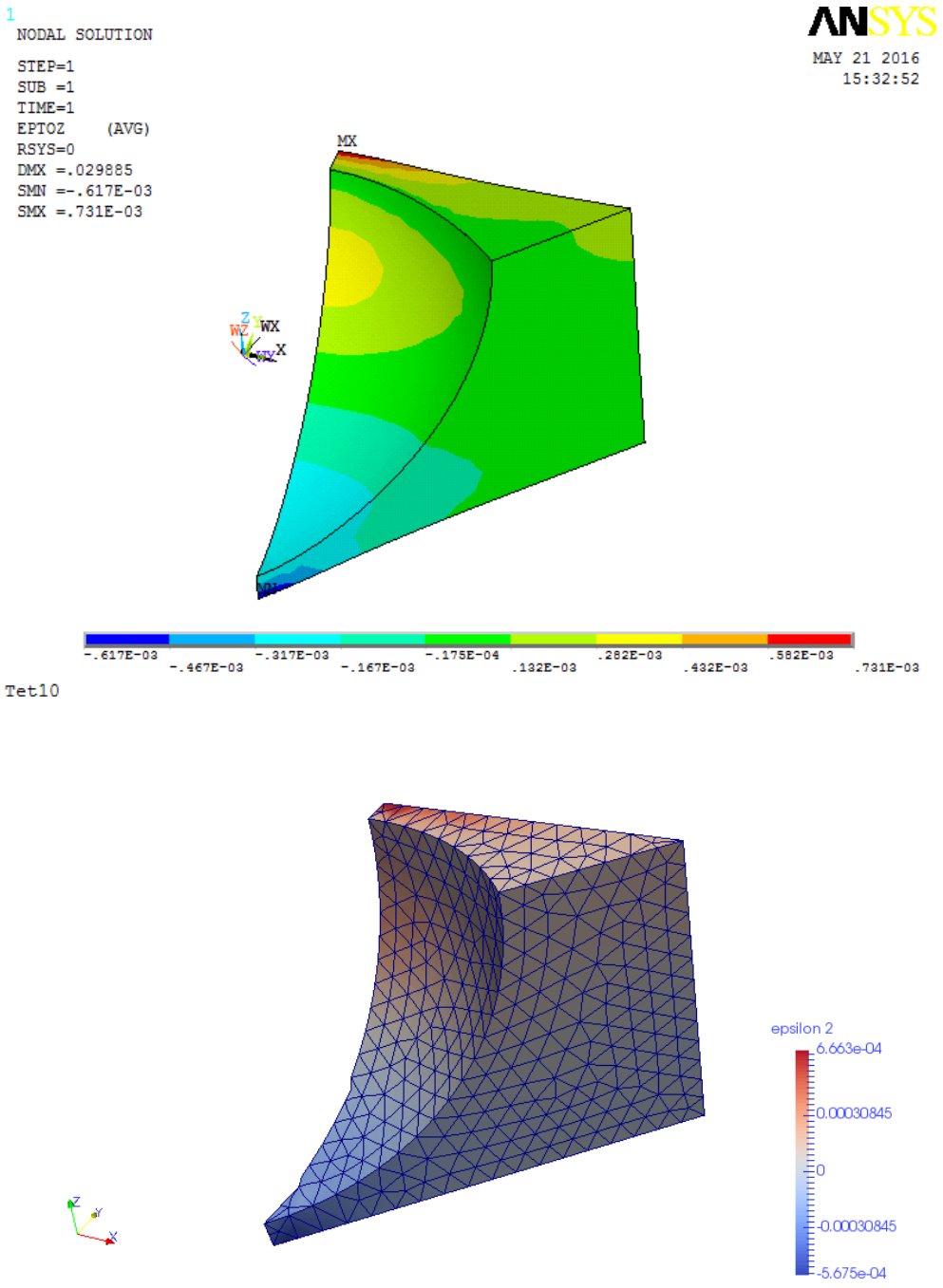


Figure 4.41: Mesh with Tet10. upper: contour plot of ϵ_{zz} from ANSYS; lower: contour plot of ϵ_{zz} calculated in AMfe, demonstrate in ParaView

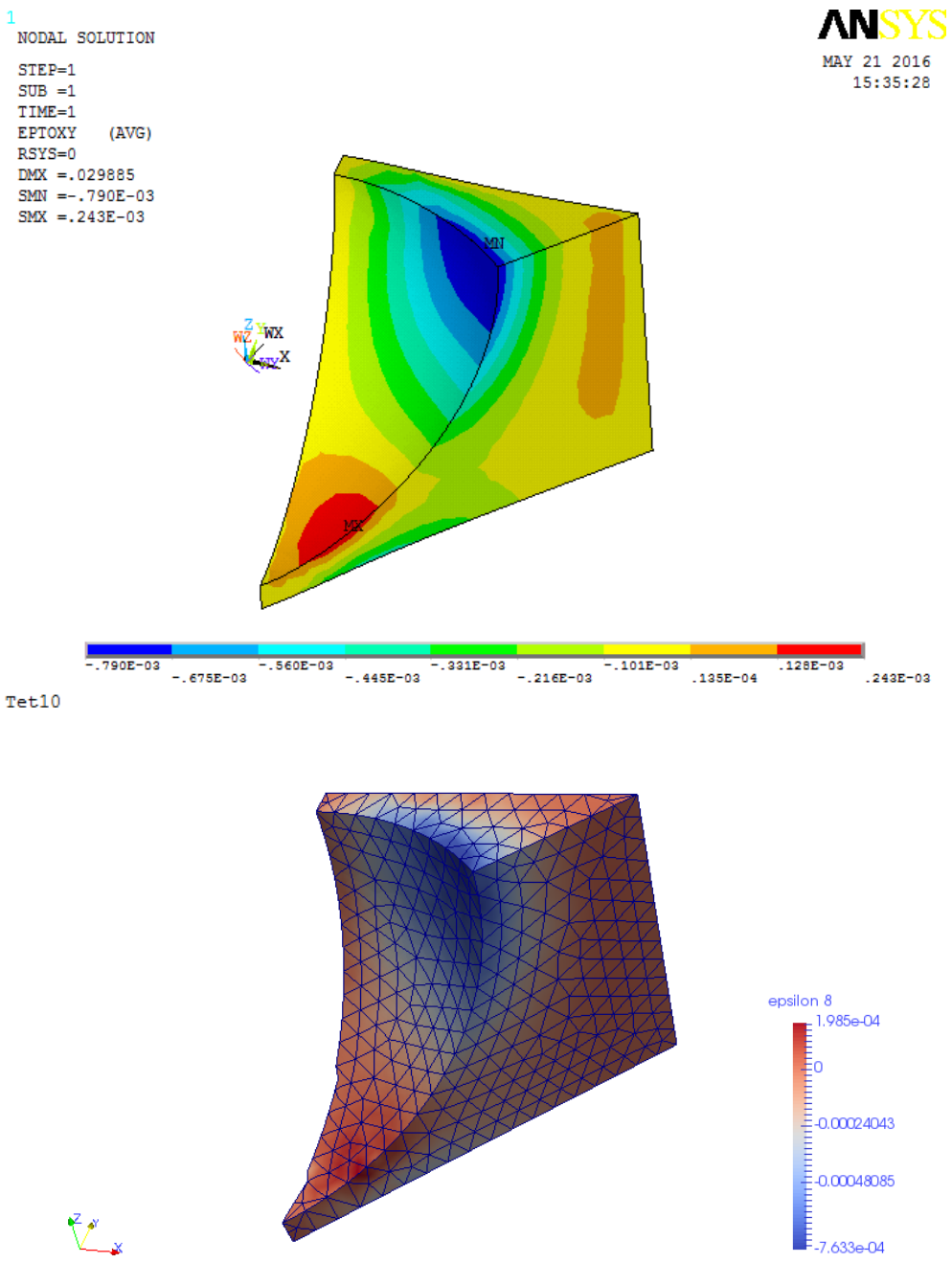


Figure 4.42: Mesh with Tet10. upper: contour plot of ϵ_{xy} from ANSYS; lower: contour plot of ϵ_{xy} calculated in AMfe, demonstrate in ParaView

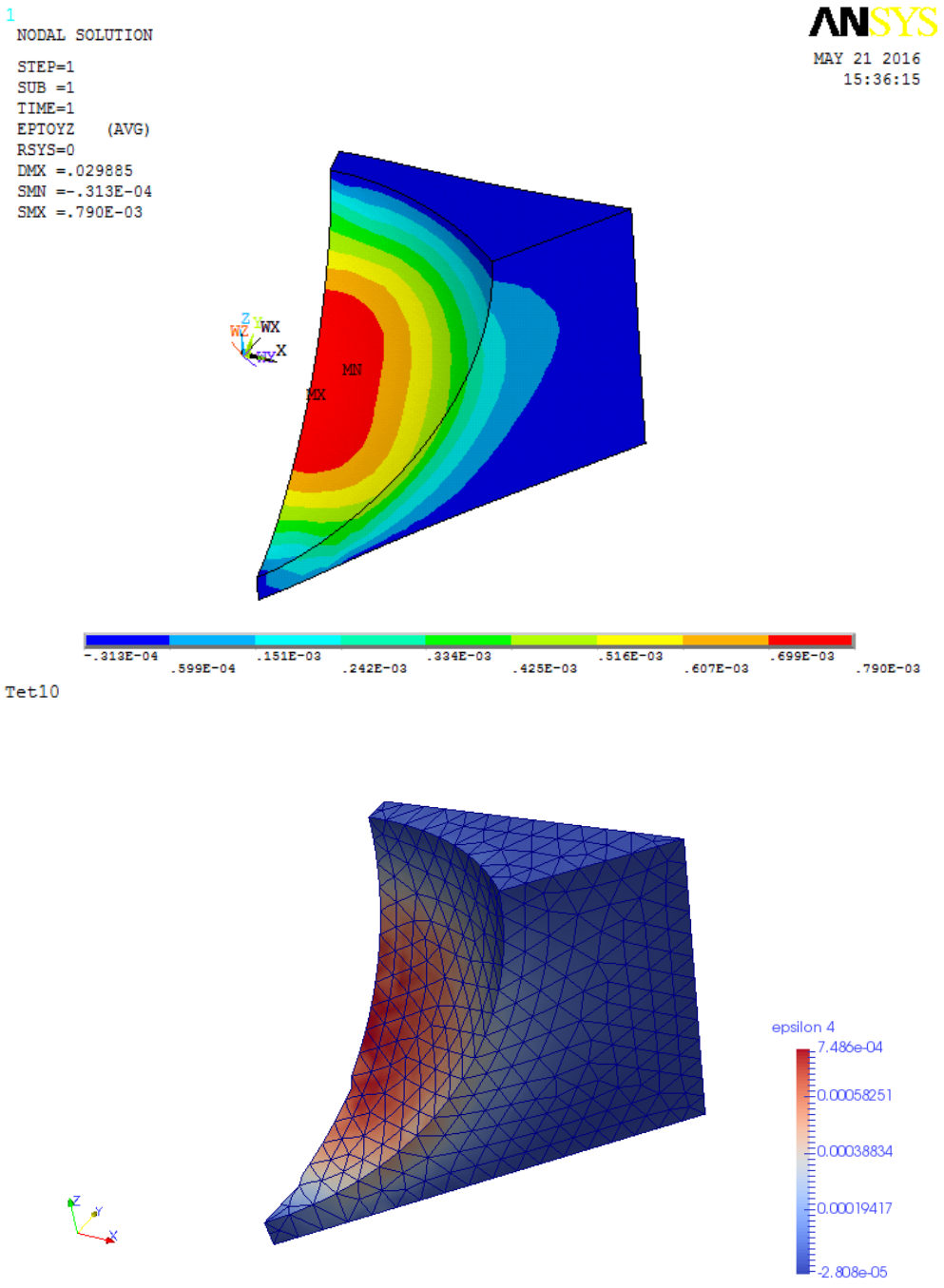


Figure 4.43: Mesh with Tet10. upper: contour plot of ϵ_{yz} from ANSYS; lower: contour plot of ϵ_{yz} calculated in AMfe, demonstrate in ParaView

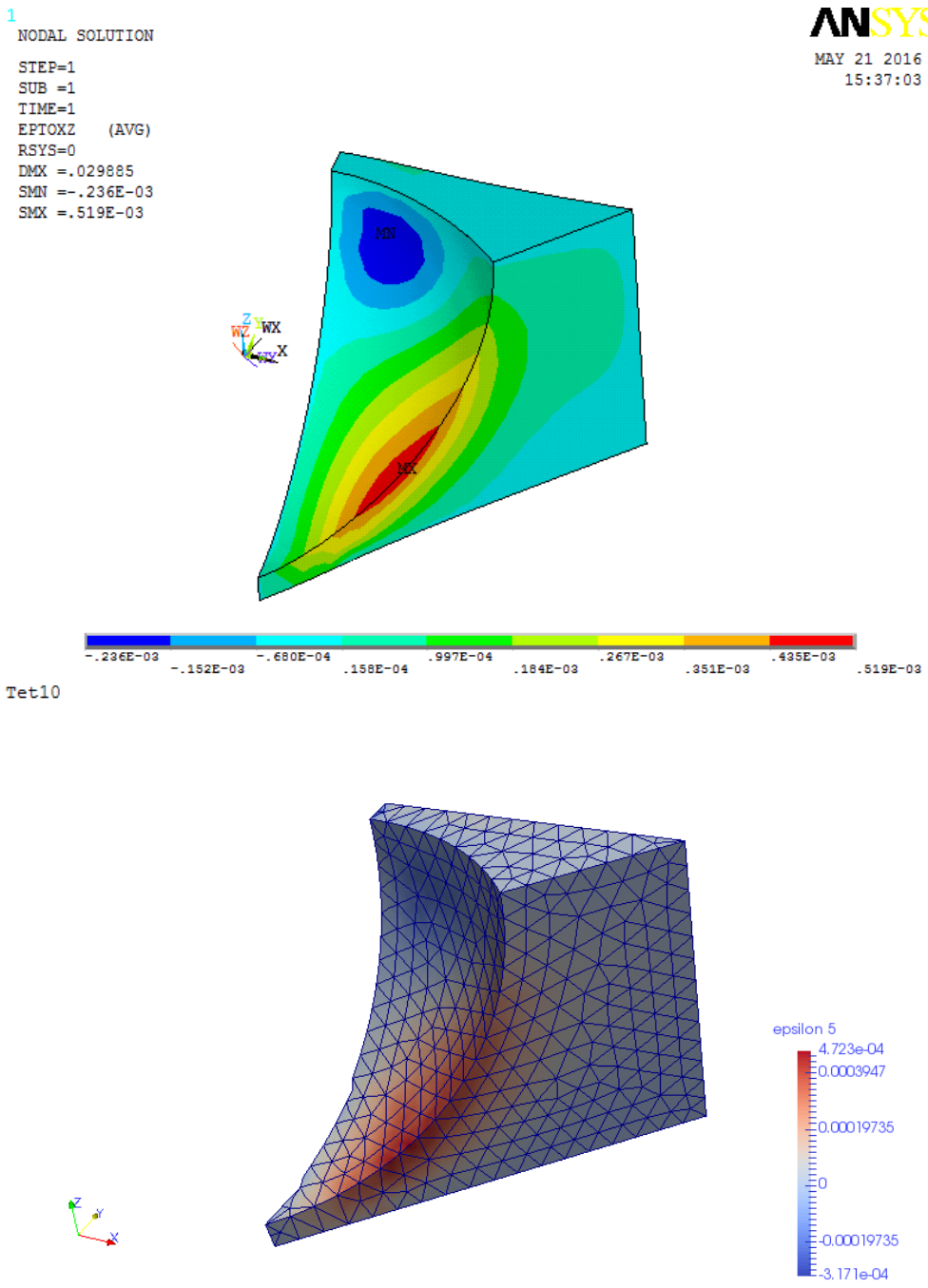


Figure 4.44: Mesh with Tet10. upper: contour plot of ϵ_{xz} from ANSYS; lower: contour plot of ϵ_{xz} calculated in AMfe, demonstrate in ParaView

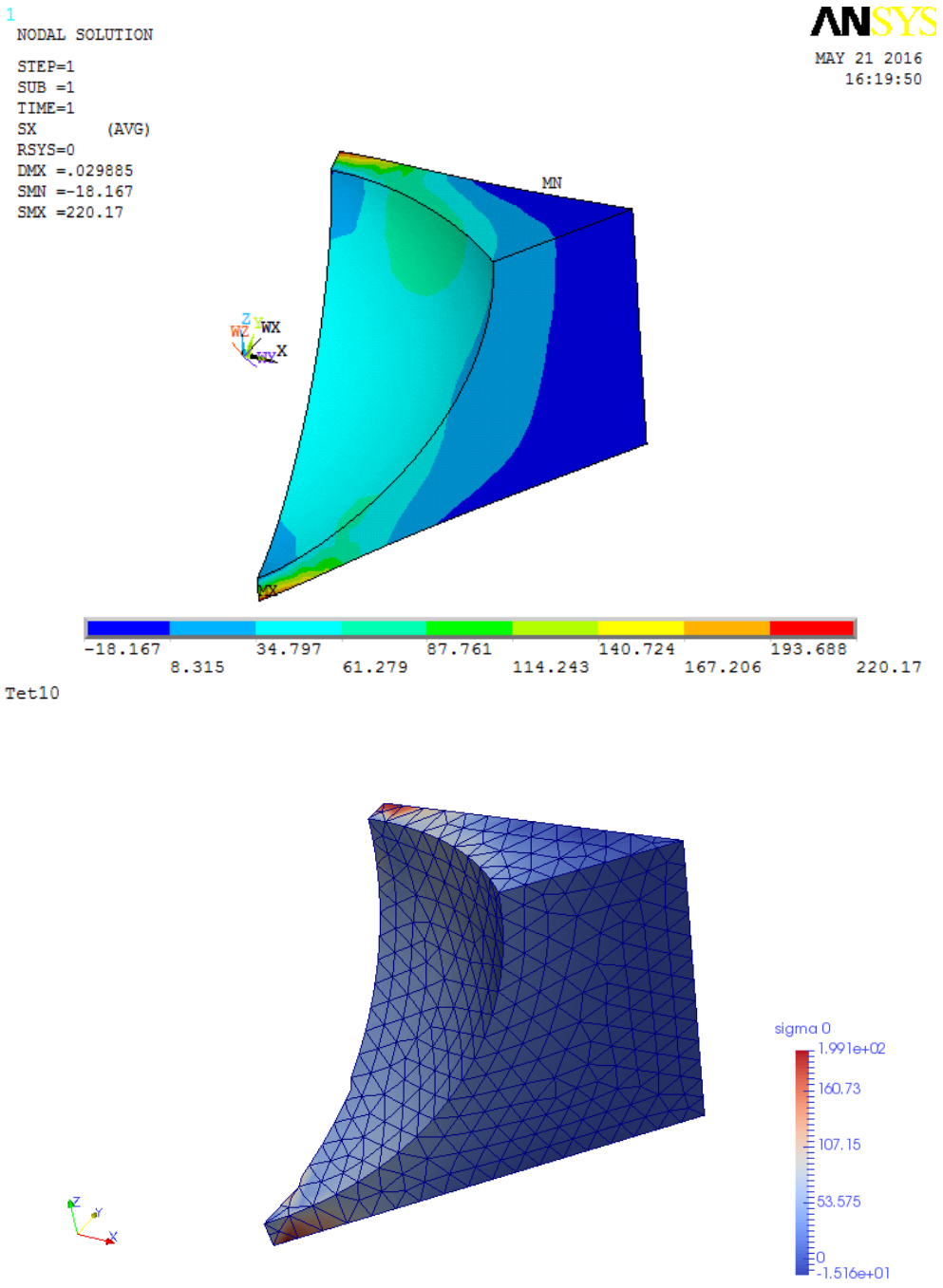


Figure 4.45: Mesh with Tet10. upper: contour plot of σ_{xx} from ANSYS; lower: contour plot of σ_{xx} calculated in AMfe, demonstrate in ParaView

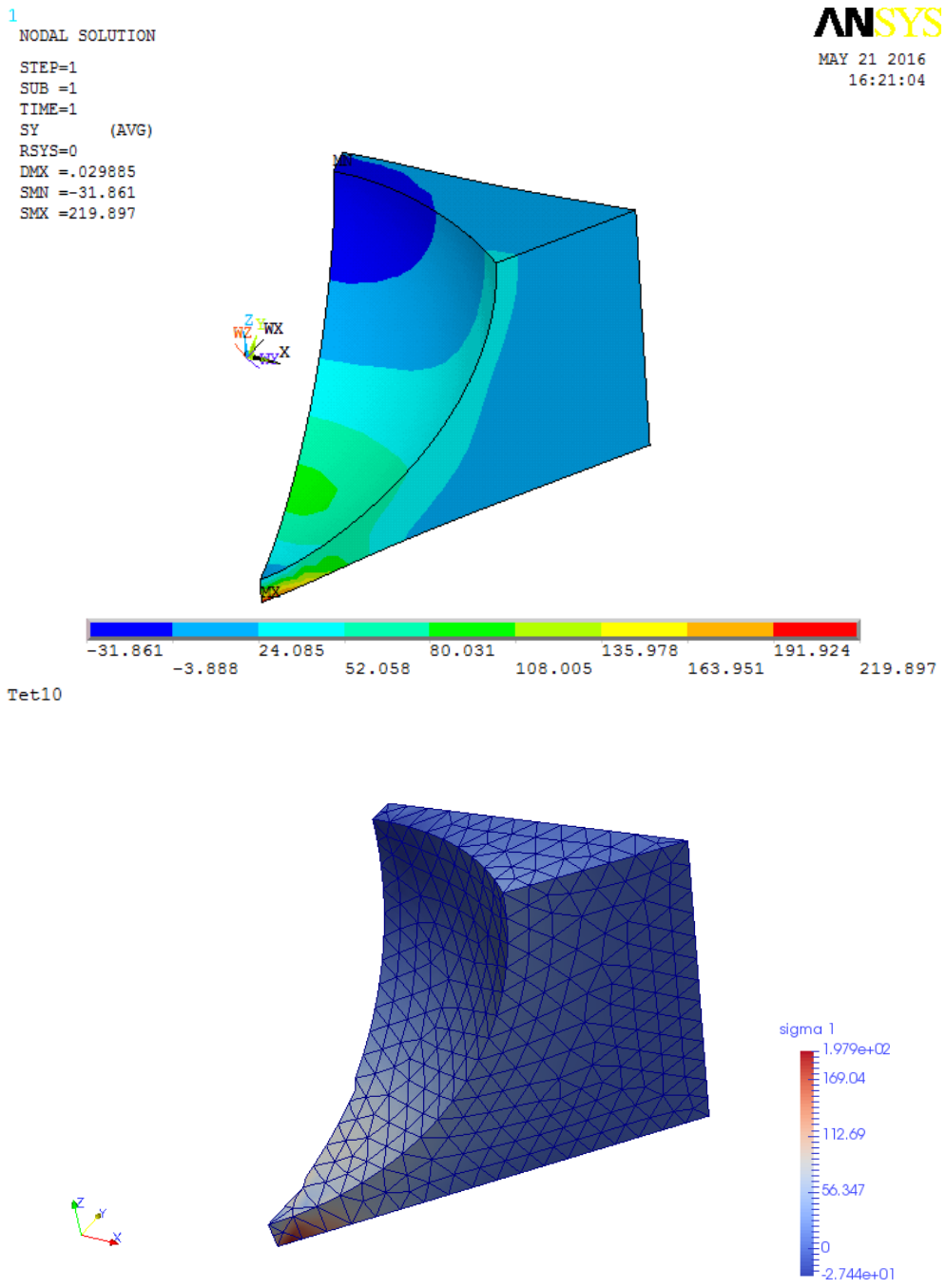


Figure 4.46: Mesh with Tet10. upper: contour plot of σ_{yy} from ANSYS; lower: contour plot of σ_{yy} calculated in AMfe, demonstrate in ParaView

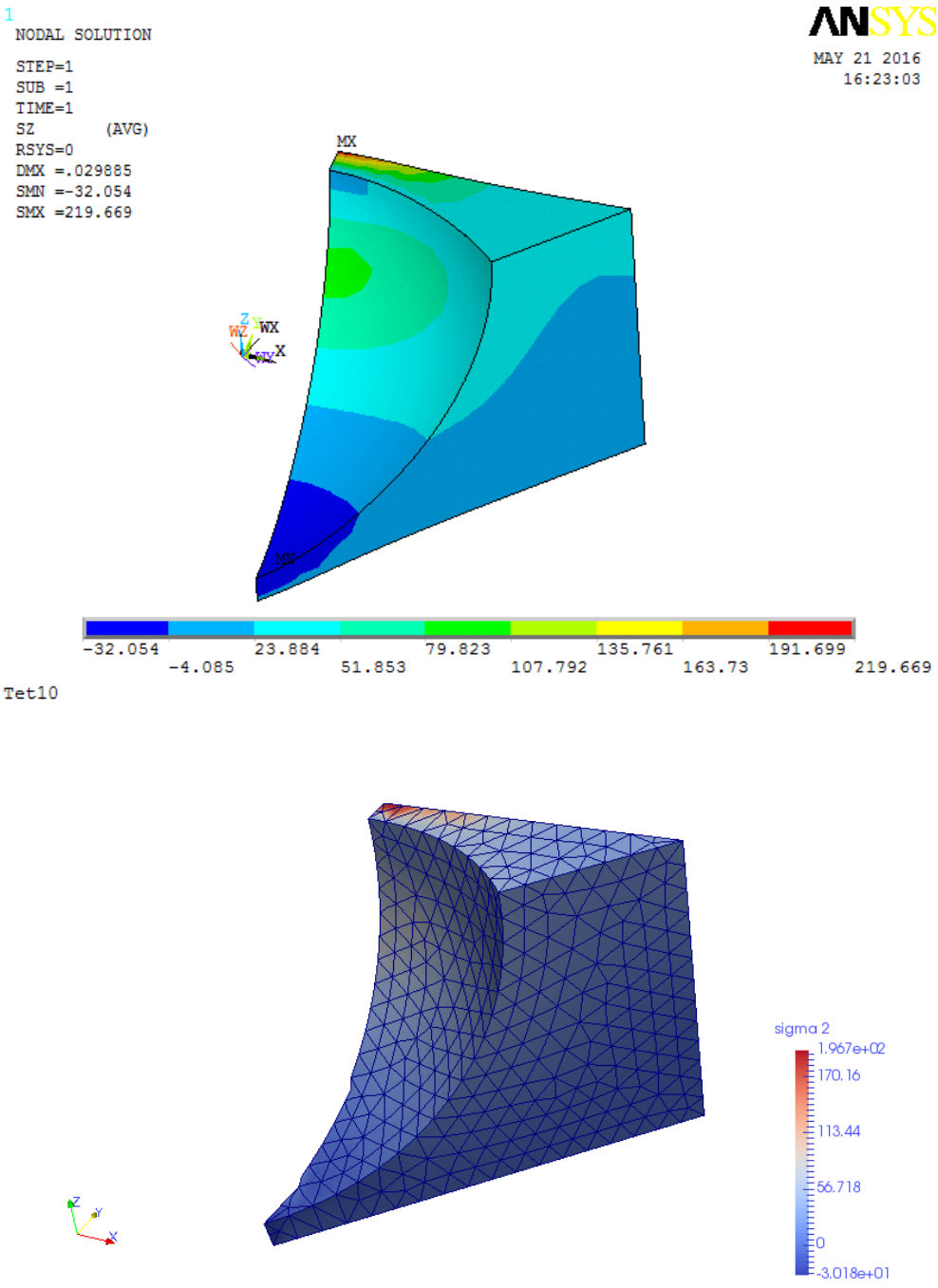


Figure 4.47: Mesh with Tet10. upper: contour plot of σ_{zz} from ANSYS; lower: contour plot of σ_{zz} calculated in AMfe, demonstrate in ParaView

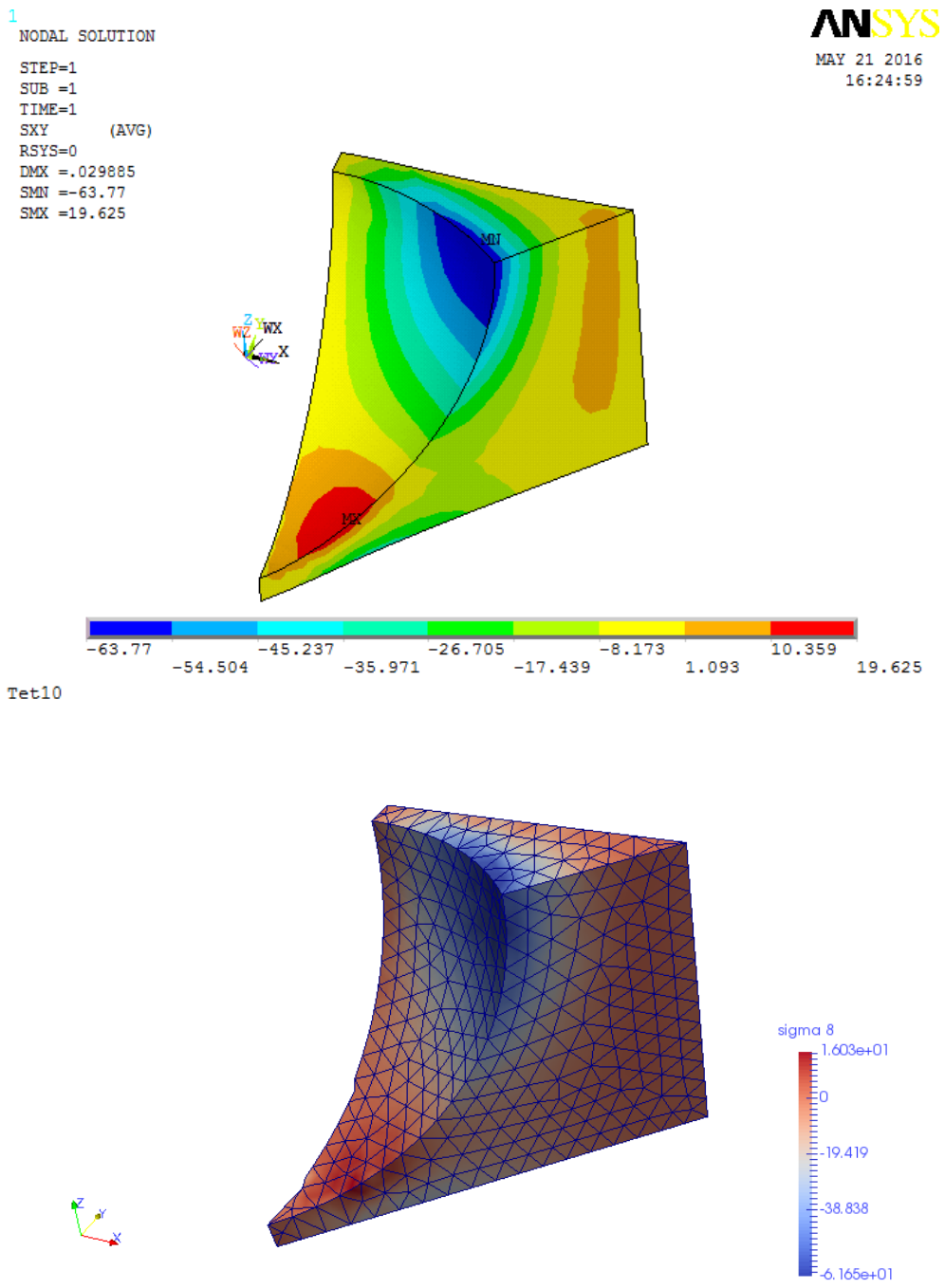


Figure 4.48: Mesh with Tet10. upper: contour plot of σ_{xy} from ANSYS; lower: contour plot of σ_{xy} calculated in AMfe, demonstrate in ParaView

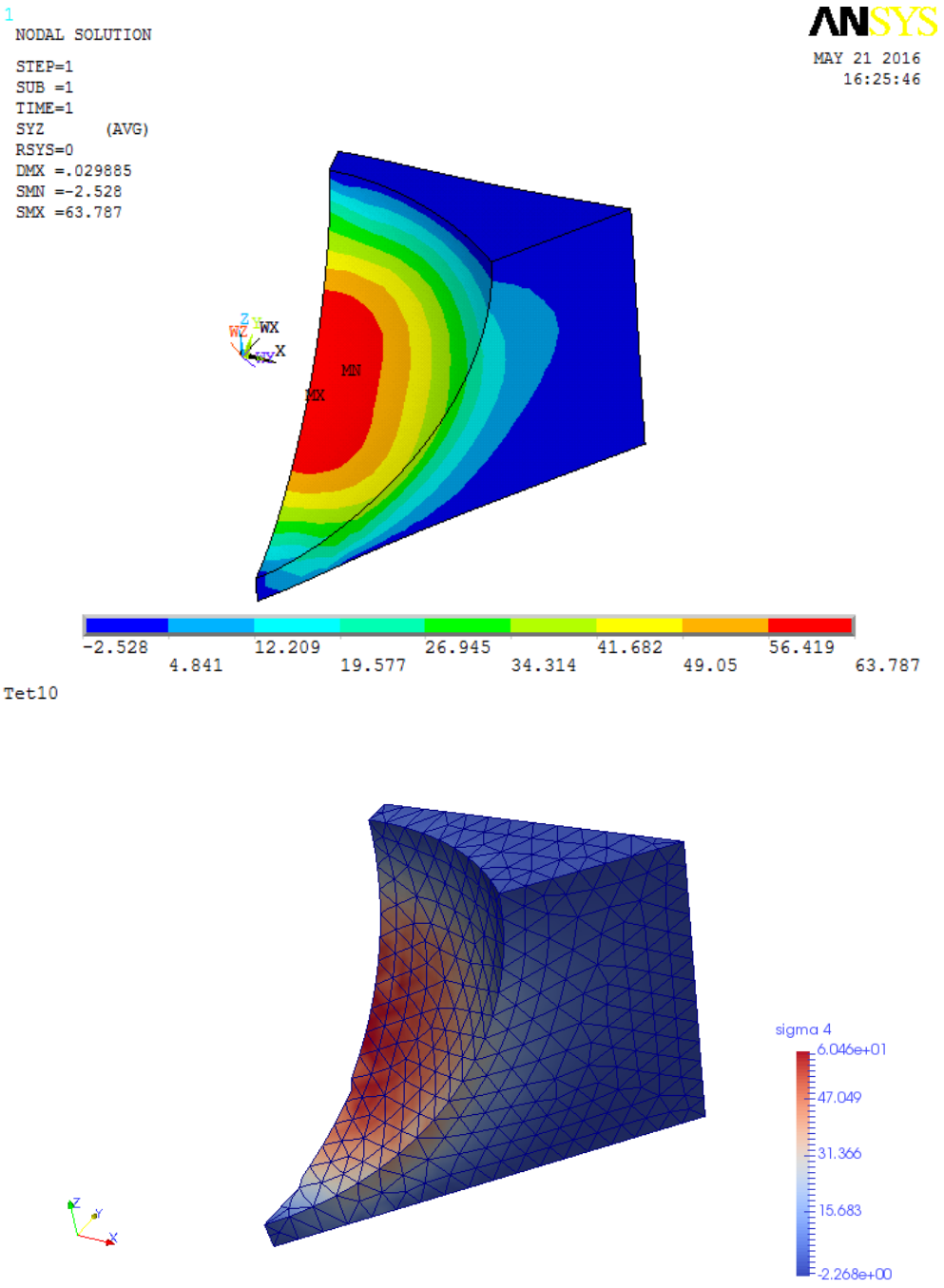


Figure 4.49: Mesh with Tet10. upper: contour plot of σ_{yz} from ANSYS; lower: contour plot of σ_{yz} calculated in AMfe, demonstrate in ParaView

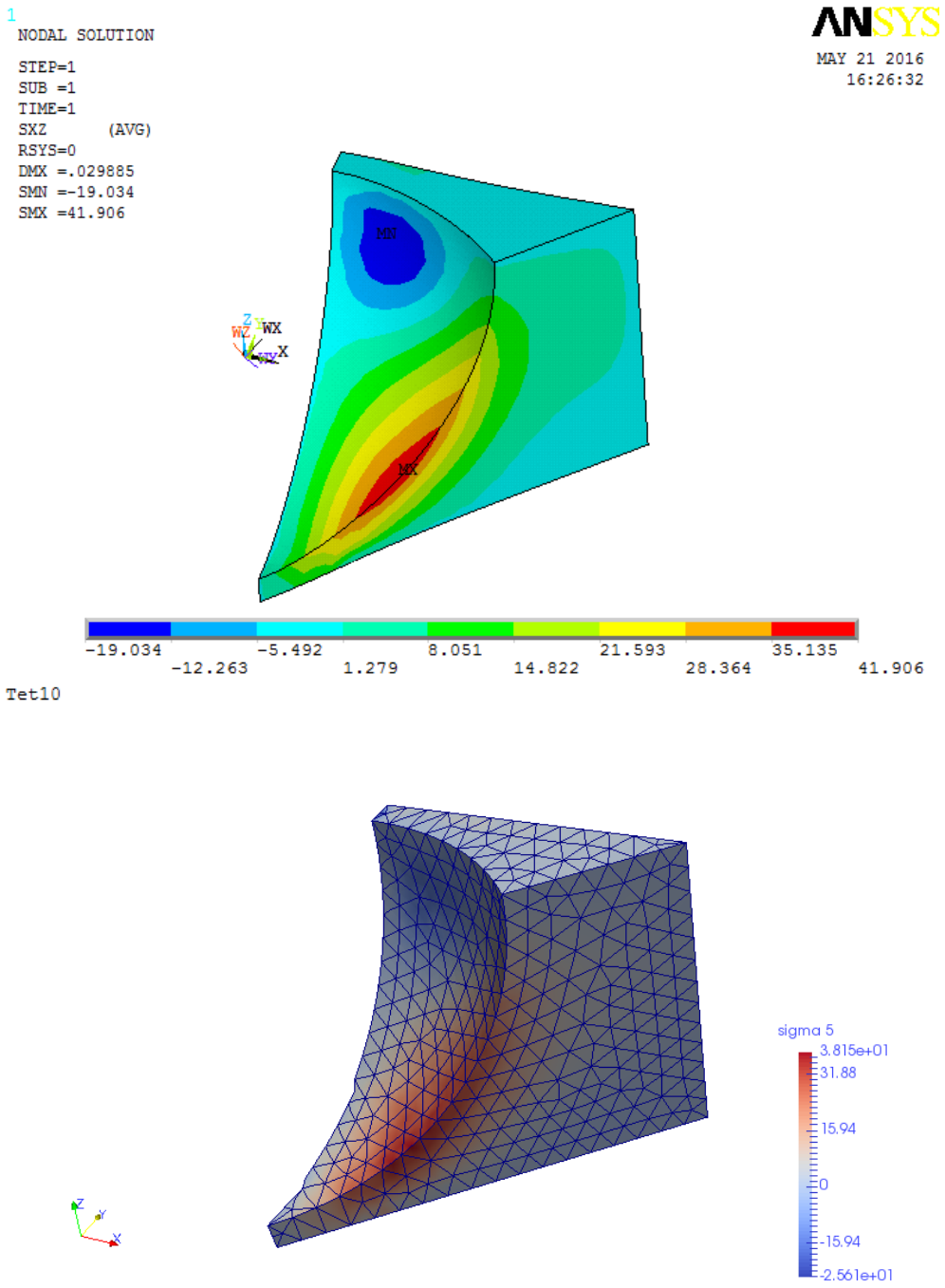


Figure 4.50: Mesh with Tet10. upper: contour plot of σ_{xz} from ANSYS; lower: contour plot of σ_{xz} calculated in AMfe, demonstrate in ParaView

4.4 Convergence Analysis

We can either decrease the element size or increase the polynomial degree of the shape function to approach an exact solution. This process is denoted as convergence. A convergence analysis provides an function to reduce the relative error of the solution with increasing number of degree of freedom [Wall 2014]. An example of convergence analysis is illustrated in Figure 4.51. Because of the technical restriction, we can only export the node, element and displacement lists from ANSYS and set all of them as input to AMfe Toolbox for solving strain and stress. We can not do the process conversely to solve the displacement by using prescribed Neumann boundary condition and Dirichlet boundary condition as input. It is still a feasible plan for future research. On the other hand, we can analyse this problem in a global point of view. It is also necessary to check the convergence of internal energy. The internal energy can be derived as:

$$\prod int = \frac{1}{2} \int_{\Omega} \sigma_{ij}^{u} \epsilon_{ij}^{u} \mathrm{d}\Omega \tag{4.3}$$

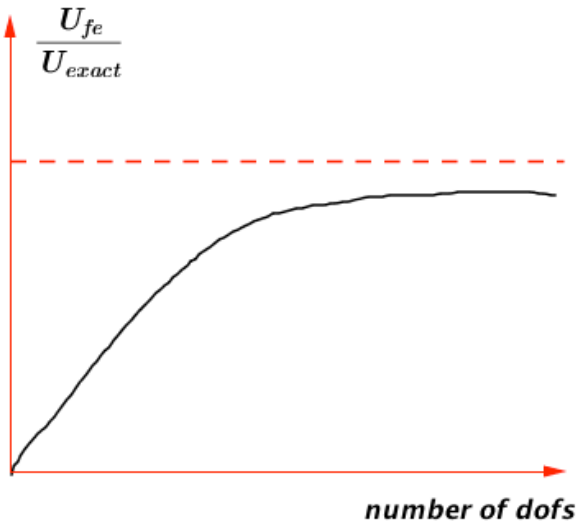


Figure 4.51: Convergence analysis

bibid Interelement Averaging can be as future research .. And I can talk about why i choose extrapolation, why not direct evaluation! Rutzmoser 2016, p. 1-10

Bibliography

Dukkipati, Rao V. (2010). *Numerical Methods*. 4835/24 Ansari Road, Daryaganj, New Delhi: New Age International Publishers.

Wall, W. A. (2014). *Finite Element*. Ed. by Svenja Schoeder. 9th ed. Lehrstuhl für Numerische Mechanik, Fakultät für Maschinenwesen, Technische Universität München.

Rutzmoser, Johannes (2016). “MEMO: Nichtlineare Finite Elemente”.

Disclaimer

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

Garching, 30. 05. 2016 _____
(Signature)