Stresses in a 10-node tetrahedron with prescribed displacements

Fino test case 050-tet10

Title	**Stresses in a 10-node tetrahedron with prescribed displacements**
Tags	elasticity stresses
Runnng time	1 sec
Available in	HTML PDF ePub

1 Problem description

Let us consider the second-order 10-node tetrahedron[1] depicted in fig. 1. The element is considered *curved* as there is at least one high-order node (i.e. the magenta ones) which is not co-linear with the two first-order (blue) nodes it links. Tbl. 2 shows the coordinates of each of the ten nodes.

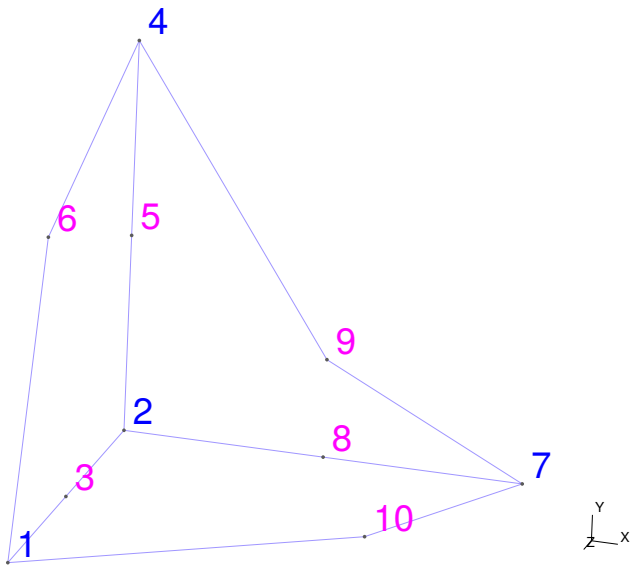


Figure 1: A ten-node curved tetrahedron.

Table 2: Index and spatial location of each of the ten nodes of the curved tetrahedron

Node index	x [m]	y [m]	z [m]
1	0.0	0.0	1.0
2	0.0	0.0	0.0
3	0.0	0.0	0.50
4	0.0	1.0	0.0

[1]The original problem was suggested by Dr. Ramsay who came up with the numbering scheme, yet the actual coordinates were later distorted in order to obtain a curved tetrahedron.

Node index	x [m]	y [m]	z [m]
5	0.0	0.50	0.0
6	0.0	0.75	0.75
7	1.0	0.0	0.0
8	0.50	0.0	0.0
9	0.50	0.25	0.0
10	0.75	0.0	0.50

The actual case is a non-physical linear elastic problem. The tetrahedron has $E = 200$ GPa and $\nu = 0.3$. Each of the ten nodes has a prescribed displacement boundary condition, namely displacement in each of the three directions numerically equal to the node index, in meters. So node 1 has $u = v = w = 1$ m, node 2 has $u = v = w = 2$ m and so on.

The matter to be answered is: what are the stresses σ_x, σ_y, σ_z, τ_{xy}, τ_{yz} and τ_{zy} evaluated at each of the ten nodes?

2 Geometry and mesh

In this case, it is better to directly create the mesh by editing the .msh file. For this particular end, version 2.2 is more human-friendly than the current version 4.1, so we use the former (Fino can read either).

```
$MeshFormat
2.2 0 8
$EndMeshFormat
$PhysicalNames
10
0 1 "p1"
0 2 "p2"
0 3 "p3"
0 4 "p4"
0 5 "p5"
0 6 "p6"
0 7 "p7"
0 8 "p8"
0 9 "p9"
0 10 "p10"
$EndPhysicalNames
$Nodes
10
        1    0.0         0.0         1.0
        2    0.0         0.0         0.0
        3    0.0         0.0        0.50
        4    0.0         1.0         0.0
        5    0.0        0.50         0.0
        6    0.0        0.75        0.75
        7    1.0         0.0         0.0
        8   0.50         0.0         0.0
        9   0.50        0.25         0.0
       10   0.75         0.0        0.50
$EndNodes
$Elements
```

```
11
1  15 2 1 1 1
2  15 2 2 2 2
3  15 2 3 3 3
4  15 2 4 4 4
5  15 2 5 5 5
6  15 2 6 6 6
7  15 2 7 7 7
8  15 2 8 8 8
9  15 2 9 9 9
10 15 2 10 10 10
11 11 2 1 1 2 7 4 1 8 9 5 3 6 10
$EndElements
```

The i-th node belongs to a physical group named "pi" over which the displacement boundary conditions will be applied in the input file.

3 Input file

The above mesh file `tet10.msh` is read, the material properties are set, the displacement conditions on each of the ten nodes are given, the problem is solved and the results written to the standard output:

```
MESH FILE_PATH tet10.msh

E = 2e11
nu = 0.3

PHYSICAL_GROUP  p1  BC u=1  v=1  w=1
PHYSICAL_GROUP  p2  BC u=2  v=2  w=2
PHYSICAL_GROUP  p3  BC u=3  v=3  w=3
PHYSICAL_GROUP  p4  BC u=4  v=4  w=4
PHYSICAL_GROUP  p5  BC u=5  v=5  w=5
PHYSICAL_GROUP  p6  BC u=6  v=6  w=6
PHYSICAL_GROUP  p7  BC u=7  v=7  w=7
PHYSICAL_GROUP  p8  BC u=8  v=8  w=8
PHYSICAL_GROUP  p9  BC u=9  v=9  w=9
PHYSICAL_GROUP  p10 BC u=10 v=10 w=10

FINO_STEP

PRINT_FUNCTION sigmax sigmay sigmaz tauxy tauyz tauzx
```

4 Results

```
$ fino tet10.fin
0    0    1    3.5122e+12    2.7627e+12    2.22102e+11   1.63814e+12   -6.90714e+09  3.67843e+11
0    0    0    6.43538e+12   4.93671e+12   3.67843e+12   2.21285e+12   8.34372e+11   1.58371e+12
0    0    0.5  4.97379e+12   3.8497e+12    1.95027e+12   1.92549e+12   4.13732e+11   9.75776e+11
0    1    0    2.19991e+12   -3.19877e+11  1.2532e+12    2.16963e+11   -2.56393e+11  1.0035e+12
0    0.5  0    4.31764e+12   2.30841e+12   2.46581e+12   1.21491e+12   2.88989e+11   1.2936e+12
0    0.75 0.75 2.85606e+12   1.22141e+12   7.37649e+11   9.27553e+11   -1.3165e+11   6.85671e+11
1    0    0    -1.90626e+11  1.57404e+12   1.66683e+12   -1.21951e+10  9.16531e+11   3.41989e+10
0.5  0    0    3.12238e+12   3.25537e+12   2.67263e+12   1.10033e+12   8.75451e+11   8.08954e+11
```

```
0.5  0.25 0    1.00464e+12   6.2708e+11    1.46001e+12   1.02384e+11   3.30069e+11   5.18849e+11
0.75 0    0.5  1.66079e+12   2.16837e+12   9.44464e+11   8.12974e+11   4.54812e+11   2.01021e+11
$
```

4.1 Check

Let us investigate what other common FEA programs say the stresses in this particular problem are.

4.1.1 CalculiX

Using the input file `tet10.inp`, CalculiX says—in its Fortran-ish output—that the stresses are

```
      1 3.51217E+12 2.76268E+12 2.22083E+11 1.63813E+12-6.91122E+09 3.67835E+11
      2 6.43535E+12 4.93669E+12 3.67841E+12 2.21284E+12 8.34369E+11 1.58370E+12
      3 4.97376E+12 3.84968E+12 1.95025E+12 1.92549E+12 4.13729E+11 9.75769E+11
      4 2.19988E+12-3.19902E+11 1.25318E+12 2.16953E+11-2.56398E+11 1.00349E+12
      5 4.31762E+12 2.30839E+12 2.46580E+12 1.21490E+12 2.88985E+11 1.29360E+12
      6 2.85603E+12 1.22139E+12 7.37631E+11 9.27543E+11-1.31654E+11 6.85664E+11
      7-1.90659E+11 1.57401E+12 1.66681E+12-1.22062E+10 9.16527E+11 3.41907E+10
      8 3.12235E+12 3.25535E+12 2.67261E+12 1.10032E+12 8.75448E+11 8.08946E+11
      9 1.00461E+12 6.27056E+11 1.45999E+12 1.02373E+11 3.30065E+11 5.18842E+11
     10 1.66076E+12 2.16835E+12 9.44446E+11 8.12964E+11 4.54808E+11 2.01013E+11
```

4.1.2 Code-Aster

Richard Szöke-Schuller kindly solved the problem using Code-Aster through Simscale. The results are

```
Node    SIXX    SIYY    SIZZ    SIXY    SIYZ    SIXZ
1   3.5698E+12  2.7837E+12  2.4351E+11  1.6544E+12  -8.7965E+09   3.8428E+11
2   6.3694E+12  4.8891E+12  3.5665E+12  2.2081E+12  8.0663E+11    1.5468E+12
3   4.9696E+12  3.8364E+12  1.9050E+12  1.9312E+12  3.9892E+11    9.6555E+11
4   2.3617E+12  -2.8201E+11 1.2429E+12  2.7308E+11  -2.8629E+11   1.0355E+12
5   4.3656E+12  2.3035E+12  2.4047E+12  1.2406E+12  2.6017E+11    1.2912E+12
6   2.9657E+12  1.2508E+12  7.4322E+11  9.6372E+11  -1.4754E+11   7.0991E+11
7   -2.8224E+11 1.3929E+12  1.2781E+12  4.0761E+09  7.8424E+11    -5.3323E+10
8   3.0436E+12  3.1410E+12  2.4223E+12  1.1061E+12  7.9544E+11    7.4674E+11
9   1.0397E+12  5.5544E+11  1.2605E+12  1.3858E+11  2.4898E+11    4.9111E+11
10  1.6438E+12  2.0883E+12  7.6080E+11  8.2922E+11  3.8772E+11    1.6548E+11
```

4.1.3 ANSYS

Angus Ramsay, who invented the problem, used non-free program ANSYS to solve it. This program does not report the stresses at the high-order nodes but only at the four first-order nodes. And yes, it seems that FORTRAN 77 with a stringent columnar format is still there too:

```
      1  0.35122E+013 0.27627E+013 0.22210E+012 0.16381E+013-0.69071E+010 0.36784E+012
      2  0.64354E+013 0.49367E+013 0.36784E+013 0.22128E+013 0.83437E+012 0.15837E+013
      4  0.21999E+013-0.31988E+012 0.12532E+013 0.21696E+012-0.25639E+012 0.10035E+013
```

```
      7 -0.19063E+012 0.15740E+013 0.16668E+013-0.12195E+011 0.91653E+012 0.34199E+011
```

4.2 Discussion

The numerical results of the six stresses as reported by Fino, CalculiX and ANSYS are equal up to the fourth decimal place. But Code-Aster gives slightly different results.

In the displacement-based finite-element formulation of the linear elastic problem, the stresses are considered secondary fields and are computed out of the spatial derivatives of the displacements, which are the primary unknown fields. In this problem, all the thirty unknown displacements (i.e. three for each of the ten nodes) are known and fixed beforehand. Any difference in the reported results is to be assigned to particular choice of one of the several possible ways of computing the nine derivatives (of each of the three displacements u, v and w with respect to each of the three coordinates x, y and z) of the primary fields.

By default, Fino computes the nine derivatives at the Gauss points and then extrapolates these values to the corner nodes. In particular, for the 10-node tetrahedron, Fino uses the four-point Gauss set so if ϕ_j^G is the value of a certain scalar field (say $\partial u/\partial x$) at the j-th Gauss point inside a certain element and ϕ_j^N is the expected value of that field at the node j within the same element, then

$$\begin{bmatrix} \phi_1^N \\ \phi_2^N \\ \phi_3^N \\ \phi_4^N \end{bmatrix} = \begin{bmatrix} h_1(c,c,c) & h_2(c,c,c) & h_3(c,c,c) & h_4(c,c,c) \\ h_1(d,c,c) & h_2(d,c,c) & h_3(d,c,c) & h_4(d,c,c) \\ h_1(c,d,c) & h_2(c,d,c) & h_3(c,d,c) & h_4(c,d,c) \\ h_1(c,c,d) & h_2(c,c,d) & h_3(c,c,d) & h_4(c,c,d) \end{bmatrix} \cdot \begin{bmatrix} \phi_1^G \\ \phi_2^G \\ \phi_3^G \\ \phi_4^G \end{bmatrix}$$

where $h_j(r,s,t)$ is the 4-node tetrahedron shape function of the j-th node in the canonical coordinates and

$$a = \frac{5-\sqrt{5}}{20}$$

$$b = \frac{5+3\sqrt{5}}{20}$$

$$c = -\frac{a}{b-a}$$

$$d = 1 + \frac{1-b}{b-a}$$

This extrapolation scheme was derived by assuming that the four Gauss points are the corner nodes of a canonical 4-node tetrahedron and then ask what would values would the field take at the four corner nodes of the original 10-node tetrahedron, all of which would have natural coordinates outside the range $[0,1]$. The same coefficients would be obtained by using other mathematical paths.

To extrapolate the values to the second-order nodes (i.e. those located in the middle of the edges) we can note that their canonical coordinates (yet not the physical ones if the element is curved) are the arithmetic mean of the ones corresponding to the first-order nodes at the end of the edge. Since the extrapolation is

a linear transformation on the canonical coordinate system, it follows that the extrapolated values at the mid-edge nodes are the average of the values at the two corner nodes.

In view of the results above, it seems that ANSYS uses either the same or a similar scheme although data for mid-edge nodes is not provided (nor the source code, so this cannot be guaranteed). CalculiX, as explained in his author's book [1], also computes nodal stresses in the same way Fino does—both for the corner and the mid-edge nodes. In this case, the source code is made available under the terms of the GPLv2+ yet the actual computation is performed within a not-very-human-friendly FORTRAN77 routine so the details could not be worked out. But in private correspondence, Dr. Dhondt stated that he coded CalculiX to compute stresses out of displacements in the same way the non-free program ABAQUS worked. In any case, once again, the results obtained with Fino, ANSYS and CalculiX coincide within a reasonable floating-point precision.

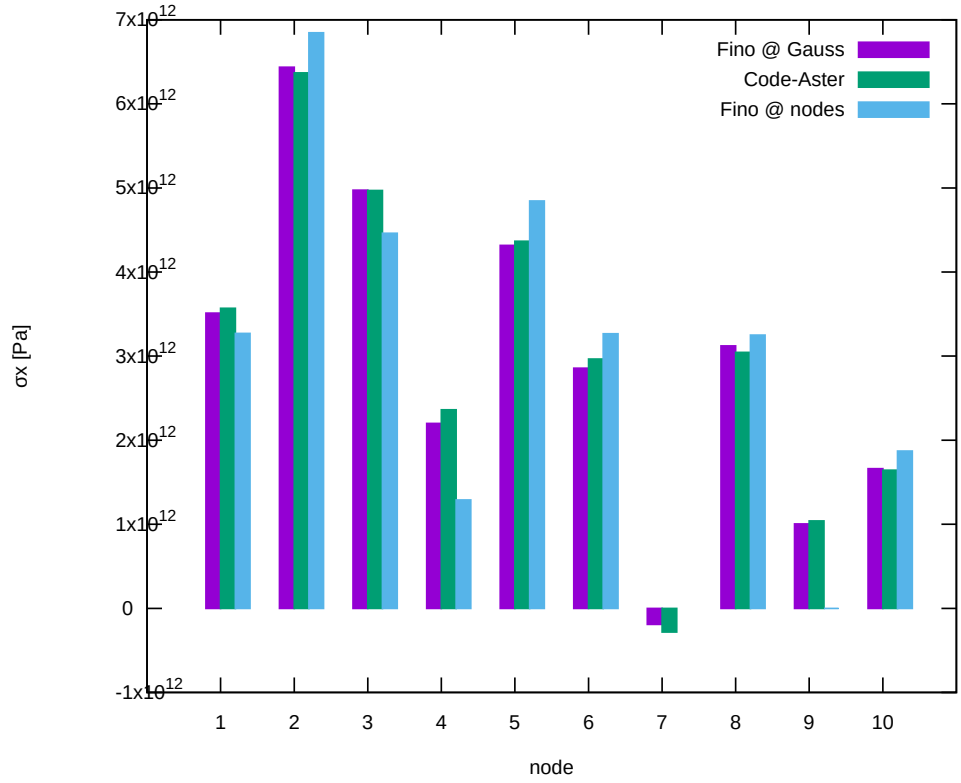
On the other hand, the Code-Aster documentation says that for 10-node tetrahedron, the derivatives are computed at a 15-point Gauss set and then extrapolated to the ten nodes using a minimum-squares projection method. It can be seen that this scheme throws different—although similar—results than the previous three.

The rationale behind the idea of computing derivatives at the Gauss points comes from the superconvergence of the derivatives of two-dimensional elliptic problems solved with a quadrangle-based mesh. Yet still another way of computing the gradient of the displacements solution would be to evaluate the derivatives of the shape functions *at* the nodes, instead of at the Gauss points and then extrapolating. Fino provides a run-time option which allows to test this scheme using the `GRADIENT` keyword. So running the same case `tet10.fin` as above but with an extra line indicating that gradients are to be computed directly at the nodes (both first and second-order ones)

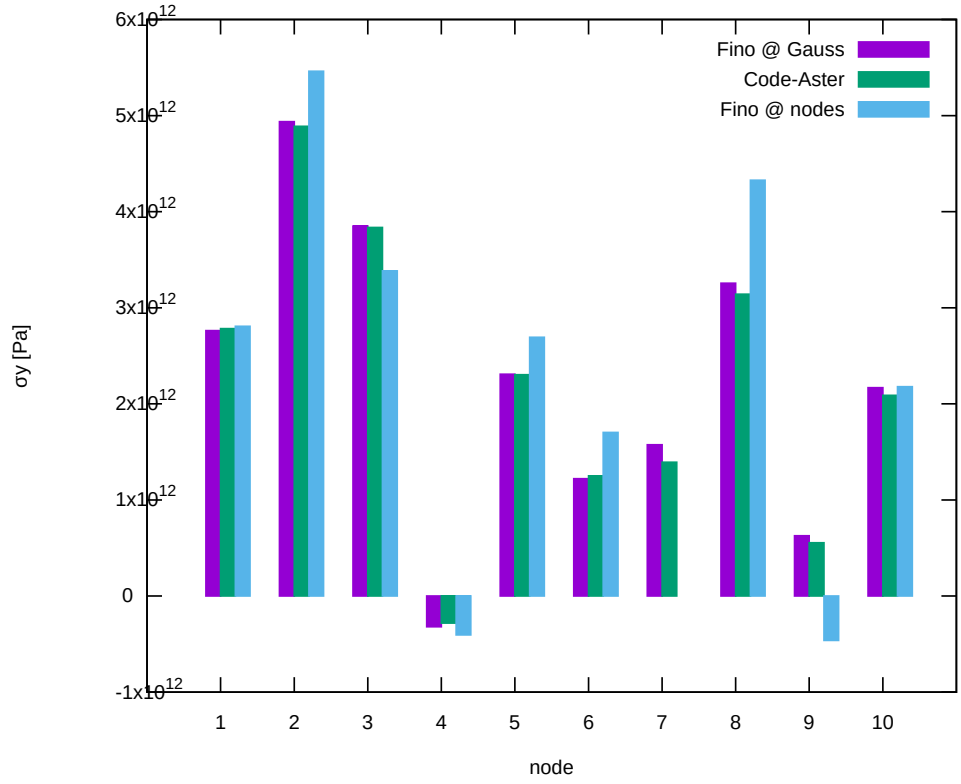
```
FINO_SOLVER GRADIENT nodes GRADIENT_HIGHER nodes
INCLUDE tet10.fin
```

```
$ fino tet10-nodes.fin
0    0    1    3.26923e+12   2.80769e+12   4.23077e+11   1.53846e+12   1.15385e+11   3.46154e+11
0    0    0    6.84615e+12   5.46154e+12   4.69231e+12   2.23077e+12   1.15385e+12   1.84615e+12
0    0    0.5  4.46154e+12   3.38462e+12   2.15385e+12   1.61538e+12   4.61538e+11   1e+12
0    1    0    1.28846e+12   -4.03846e+11  1.36538e+12   -7.69231e+10  -3.84615e+10  8.07692e+11
0    0.5  0    4.84615e+12   2.69231e+12   2.79487e+12   1.38462e+12   3.58974e+11   1.4359e+12
0    0.75 0.75 3.26603e+12   1.70192e+12   1.24038e+12   1.05128e+12   3.84615e+10   8.20513e+11
1    0    0    -nan    -nan  -nan    -nan  -nan    -nan
0.5  0    0    3.25e+12      4.32692e+12   3.17308e+12   1.30769e+12   1.26923e+12   7.30769e+11
0.5  0.25 0    9.15527e-05   -4.61538e+11  1.12821e+12   -3.84615e+11  1.79487e+11   4.10256e+11
0.75 0    0.5  1.87179e+12   2.17949e+12   9.48718e+11   8.71795e+11   4.10256e+11   2.5641e+11
```

This tetrahedron has the particularity that the jacobian of the transformation between the real physical x-y-z and the canonical r-s-t coordinates is singular at the node number 7. Hence, direct evaluation fails. The following figures compare the 4-point-Gauss-extrapolated results (i.e. Fino with the default options, CalculiX and presumably ANSYS if it could write stresses at all nodes) with the 15-point to 10-point minimum squares projection by CodeAster with direct evaluation of the derivatives at the nodes by Fino.

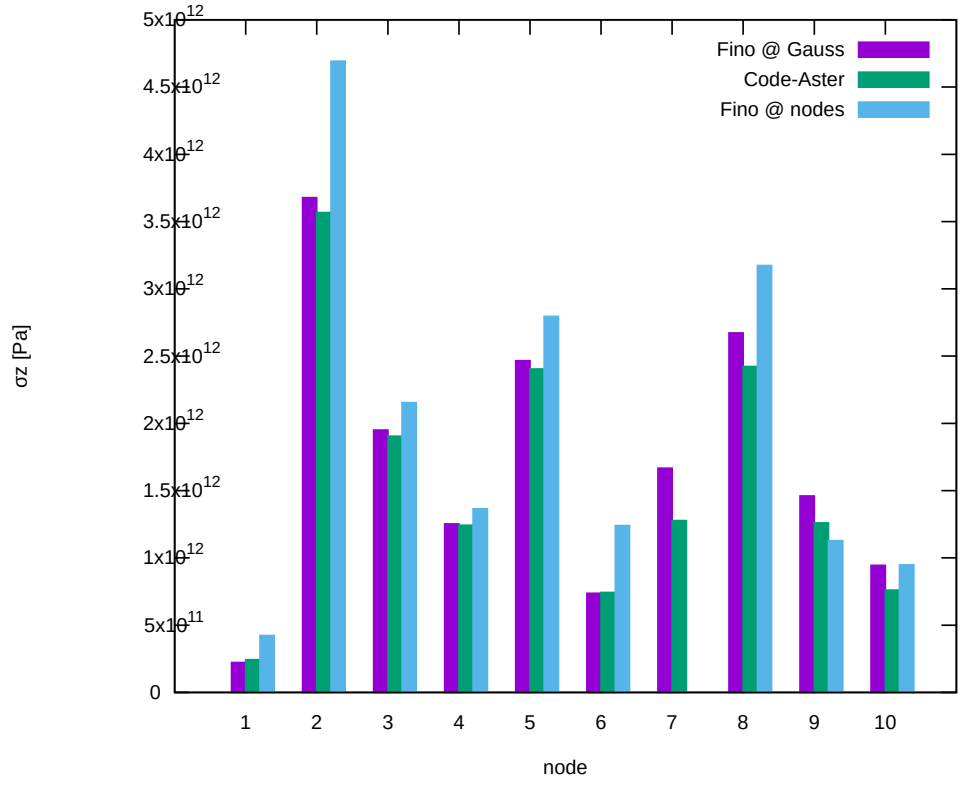


(a) σ_x

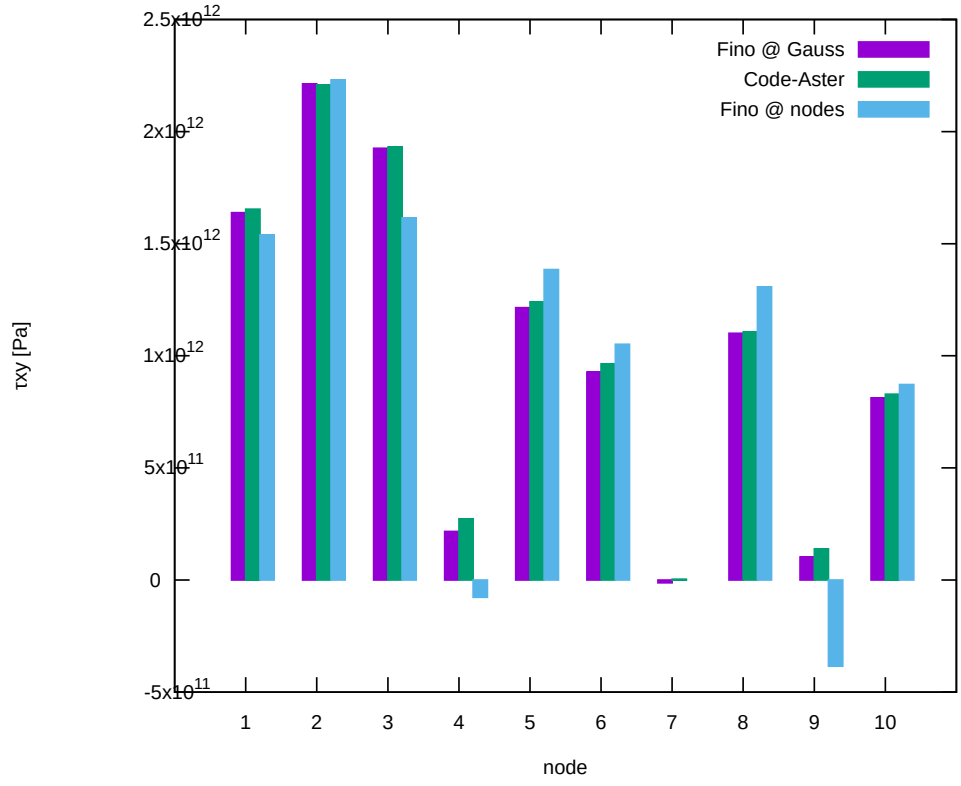


(b) σ_y

Figure 2: Comparison of stresses at each node (1/3)

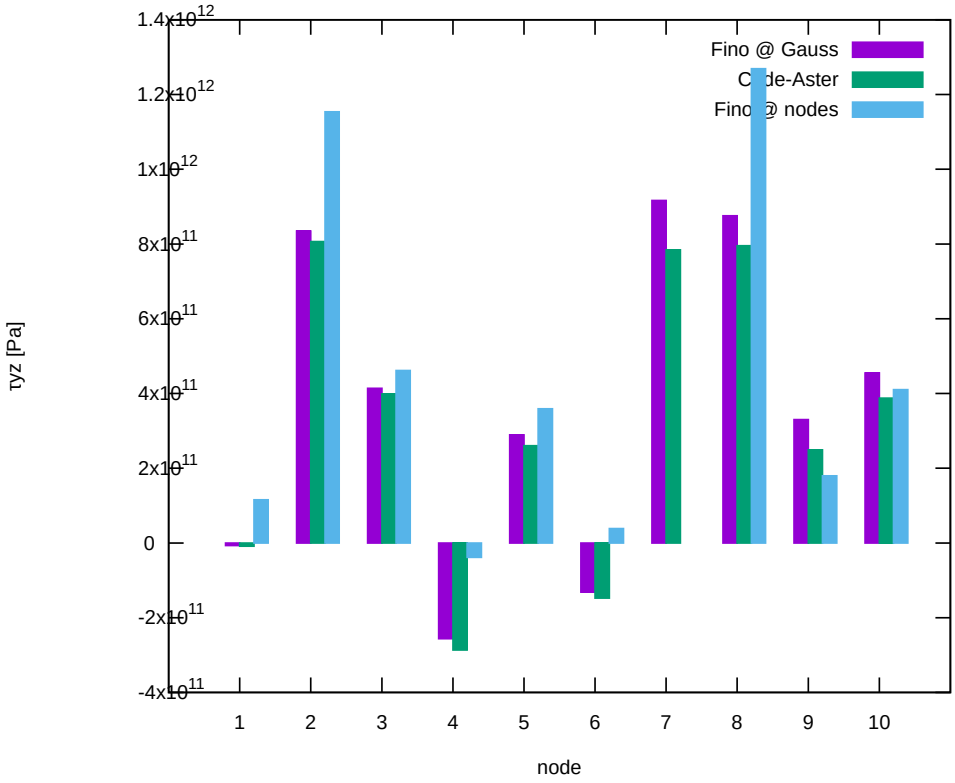


(a) σ_z

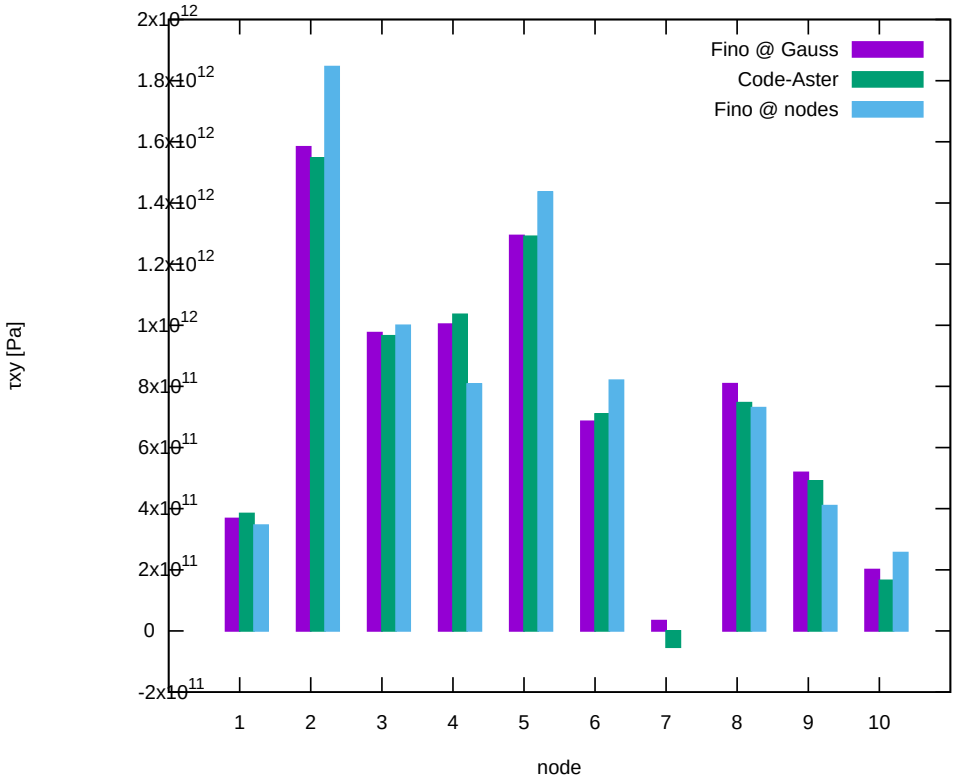


(b) τ_{xy}

Figure 3: Comparison of stresses at each node (2/3)



(a) τ_{yz}



(b) τ_{zx}

Figure 4: Comparison of stresses at each node (2/3)

It should be noted though that if the tetrahedron was not "curved" but "straight" (i.e the mid-edge nodes were located in the straight line that connects the corner nodes), both the Gauss-extrapolated values and the direct evaluation at the nodes would give the same results. It is expected the differences observed in the figures above decrease with increasing element quality and decreasing size. In effect, if we replaced the node coordinates with, say

```
$Nodes
10
        1    0.0         0.0         1.0
        2    0.0         0.0         0.0
        3    0.0         0.0         0.5
        4    0.0         0.5         0.0
        5    0.0         0.25        0.0
        6    0.0         0.25        0.5
        7    0.25        0.0         0.0
        8    0.125       0.0         0.0
        9    0.125       0.25        0.0
       10    0.125       0.0         0.50
$EndNodes
```

and use this input file for Fino which reads the parameters for `GRADIENT` and `GRADIENT_HIGHER` from the command line

```
FINO_SOLVER GRADIENT $1 GRADIENT_HIGHER $2
MESH FILE_PATH straight.msh

E = 2e11
nu = 0.3

PHYSICAL_GROUP  p1  BC u=1  v=1  w=1
PHYSICAL_GROUP  p2  BC u=2  v=2  w=2
PHYSICAL_GROUP  p3  BC u=3  v=3  w=3
PHYSICAL_GROUP  p4  BC u=4  v=4  w=4
PHYSICAL_GROUP  p5  BC u=5  v=5  w=5
PHYSICAL_GROUP  p6  BC u=6  v=6  w=6
PHYSICAL_GROUP  p7  BC u=7  v=7  w=7
PHYSICAL_GROUP  p8  BC u=8  v=8  w=8
PHYSICAL_GROUP  p9  BC u=9  v=9  w=9
PHYSICAL_GROUP  p10 BC u=10 v=10 w=10

FINO_STEP

PRINT_FUNCTION sigmax sigmay sigmaz tauxy tauyz tauzx
```

then

```
$ fino straight.fin gauss average > default.txt
$ fino straight.fin nodes nodes > nodes.txt
$ diff default.txt nodes.txt
9c9
< 0.125 0.25   0   1.42308e+12   1.92308e+11   1.88462e+12   -0.00157139   2.30769e+11   8.46154e+11
---
> 0.125 0.25   0   1.42308e+12   1.92308e+11   1.88462e+12   0             2.30769e+11   8.46154e+11
```

```
$
```

The reported relative difference is 10^{-15} which is the order of magnitude of the double-precision floating point representation of such stresses in a digital computer.

[1] G. Dhondt, *The finite element method for three-Dimensional thermomechanical applications.* John Wiley & Sons, 2004.