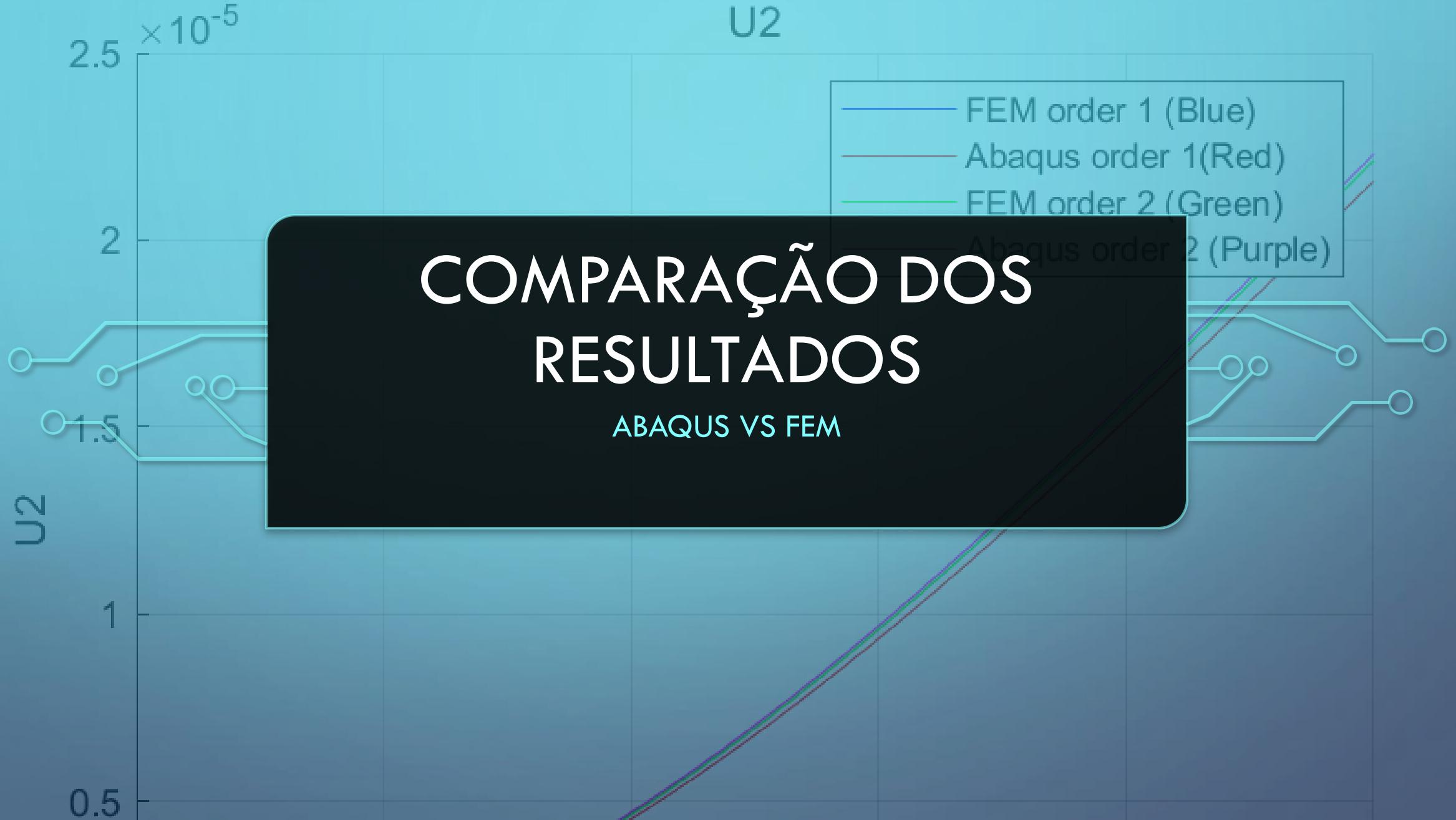


Bending

U2

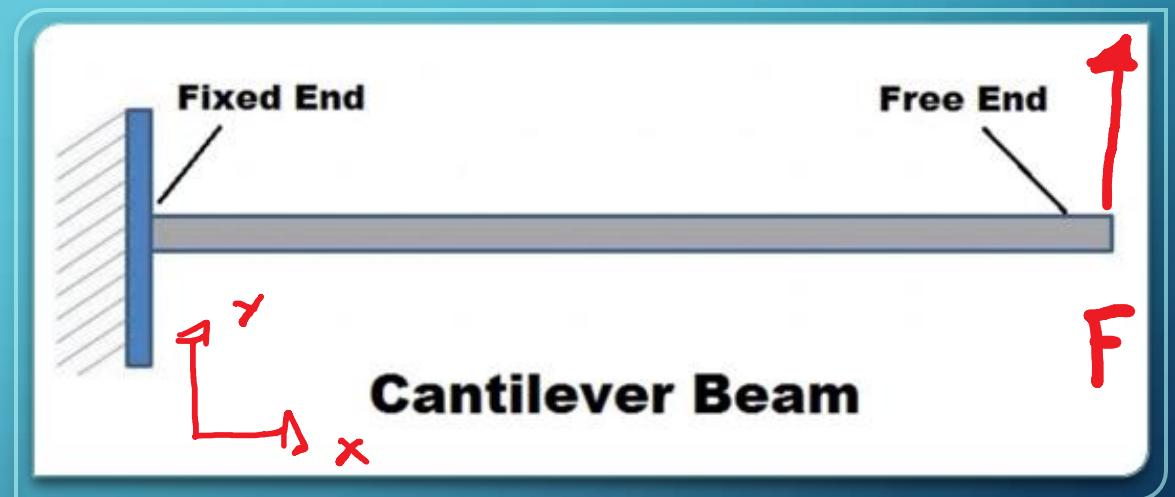


COMPARAÇÃO DOS RESULTADOS

ABAQUS VS FEM

PARÂMETROS

- Comprimento = 10 [m]
- Altura = 1 [m]
- $F = 11e2$ [N]
- Elementos em $x = 40$
- Elementos em $y = 10$



```

case 2
for iel = 1:size(obj.Matrix.Pre.Mesh.Connectivity, 1) % for each element
    inode = 1; % for each node in the element
    for iwy = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in y
        for iwx = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in x
            for icomp = 1:(obj.Matrix.Pre.Mesh.Dim+(obj.Matrix.Pre.Mesh.Dim^2-obj.Matrix.Pre.Mesh.Dim)/2) % for each component
                index = sort(obj.Matrix.Pre.Mesh.Connectivity(iel, 2:end));
                e(index, icomp) = e(index, icomp) + obj.E(iel, iwx, iwy, icomp);
                s(index, icomp) = s(index, icomp) + obj.S(iel, iwx, iwy, icomp);

```

```
mis(index, 1) = mis(index, 1) + obj.Mises(iel, iwx, iwy);  
med(index, 1) = med(index, 1) + 1;  
inode = inode + 1;
```

RESULTADOS

DESLOCAMENTO

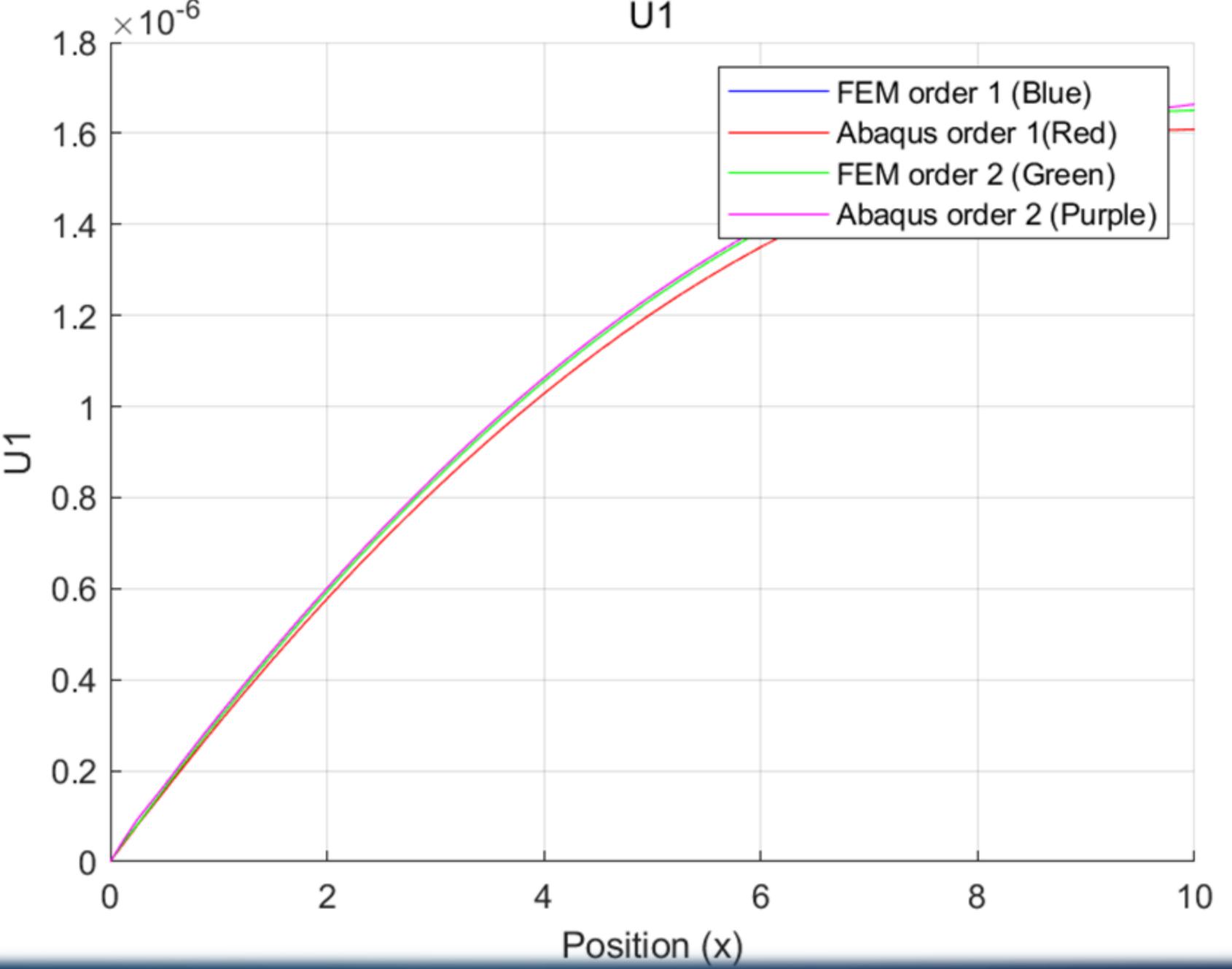
```

for iel = 1:size(obj.Matrix.Pre.Mesh.Connectivity, 1) % for each element
    inode = 1; % for each node in the element
    for iwz = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in z
        for iwy = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in y
            for iwx = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in x
                for icomp = 1:(obj.Matrix.Pre.Mesh.Dim+(obj.Matrix.Pre.Mesh.Dim^2-obj.Matrix.Pre.Mesh.Dim)/2)
                    index = sort(obj.Matrix.Pre.Mesh.Connectivity(iel, 2:end));
                    e(index, icomp) = e(index, icomp) + obj.E(iel, iwx, iwy, iwz, icomp);
                    s(index, icomp) = s(index, icomp) + obj.S(iel, iwx, iwy, iwz, icomp);
                end
                mis(index, 1) = mis(index, 1) + obj.Mises(iel, iwx, iwy, iwz);
                med(index, 1) = med(index, 1) + 1;
                inode = inode + 1;
            end
        end
    end
end

```

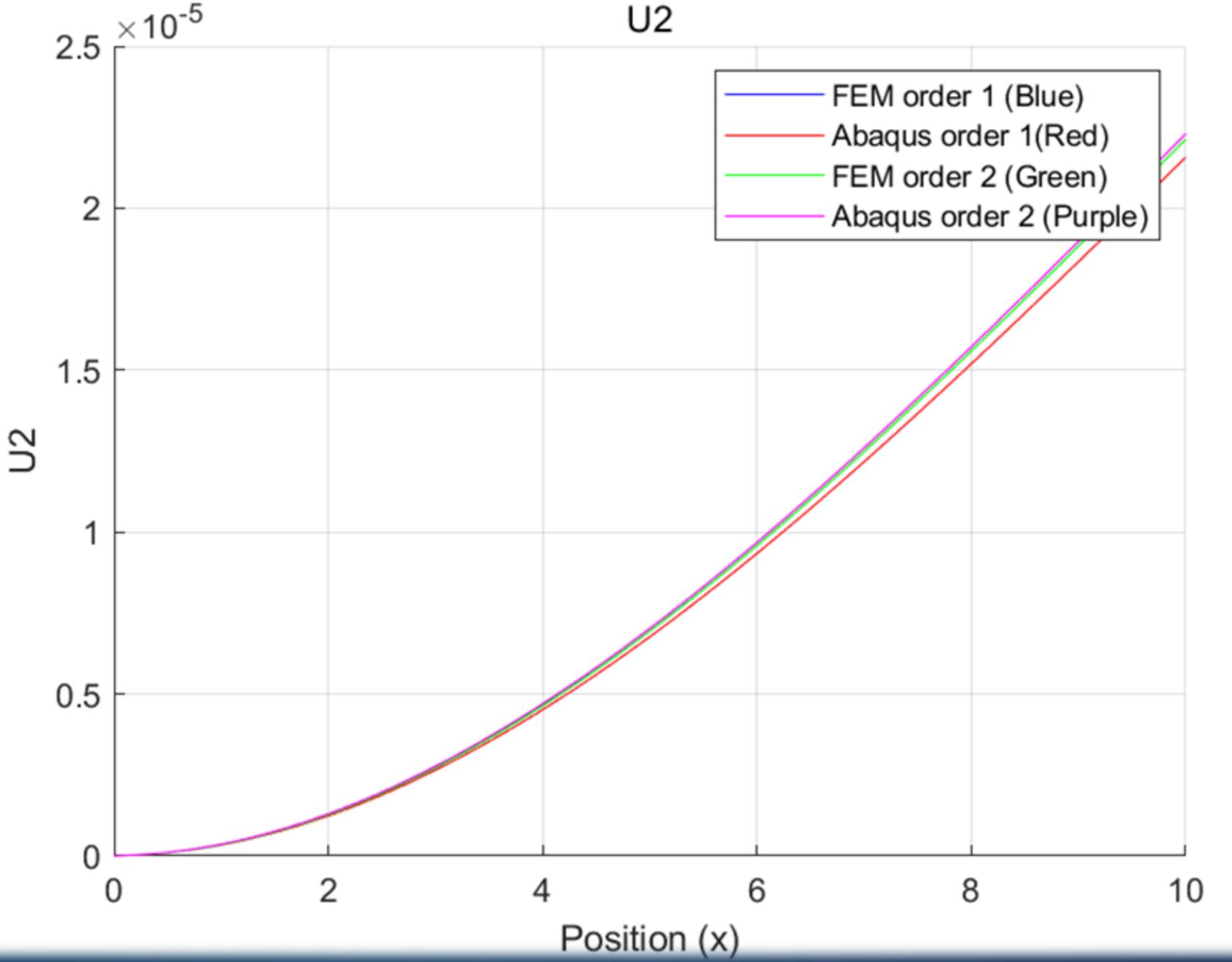
Bending

U1



Bending

U2



```
case 2
    for iel = 1:size(obj.Matrix.Pre.Mesh.Connectivity, 1) % for each element
        inode = 1; % for each node in the element
        for iwy = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in y
            for iwx = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in x
                for icomp = 1:(obj.Matrix.Pre.Mesh.Dim+(obj.Matrix.Pre.Mesh.Dim^2-obj.Matrix.Pre.Mesh.Dim)/2) % for each comp
```

```
                index = sort(obj.Matrix.Pre.Mesh.Connectivity(iel, 2:end));
```

```
                e(index, icomp) = e(index, icomp) + obj.E(iel, iwx, iwy, icomp);
```

```
                s(index, icomp) = s(index, icomp) + obj.S(iel, iwx, iwy, icomp);
```

```
            end
```

```
            mis(index, 1) = mis(index, 1) + obj.Mises(iel, iwx, iwy);
```

```
            med(index, 1) = med(index, 1) + 1;
```

```
            inode = inode + 1;
```

```
        end
```

```
    end
```

```
end
```

```
case 3
```

```
for iel = 1:size(obj.Matrix.Pre.Mesh.Connectivity, 1) % for each element
```

```
    inode = 1; % for each node in the element
```

```
    for iwz = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in z
```

```
        for iwy = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in y
```

```
            for iwx = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in x
```

```
                for icomp = 1:(obj.Matrix.Pre.Mesh.Dim+(obj.Matrix.Pre.Mesh.Dim^2-obj.Matrix.Pre.Mesh.Dim)/2) % for each comp
```

```
                    index = sort(obj.Matrix.Pre.Mesh.Connectivity(iel, 2:end));
```

```
                    e(index, icomp) = e(index, icomp) + obj.E(iel, iwx, iwy, iwz, icomp);
```

```
                    s(index, icomp) = s(index, icomp) + obj.S(iel, iwx, iwy, iwz, icomp);
```

```
                end
```

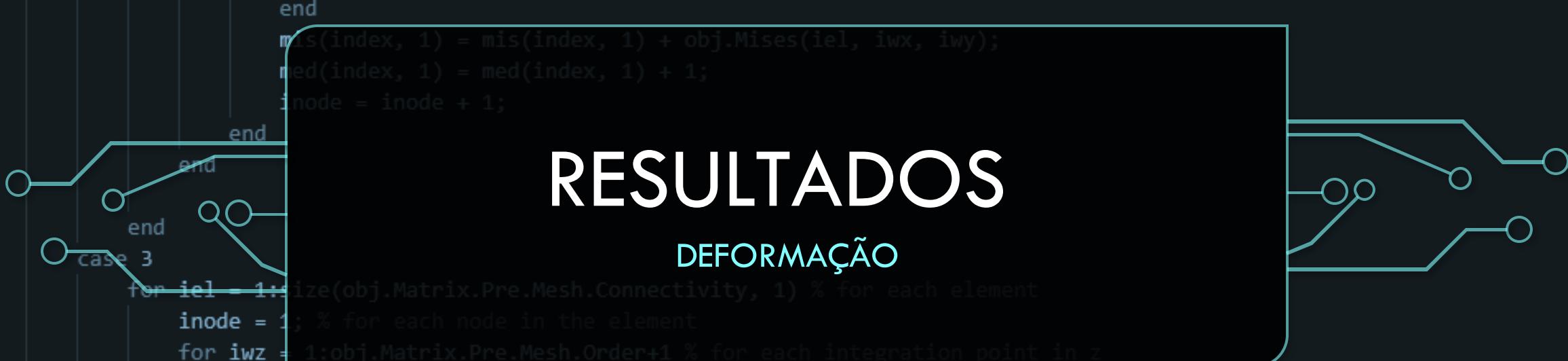
```
                mis(index, 1) = mis(index, 1) + obj.Mises(iel, iwx, iwy, iwz);
```

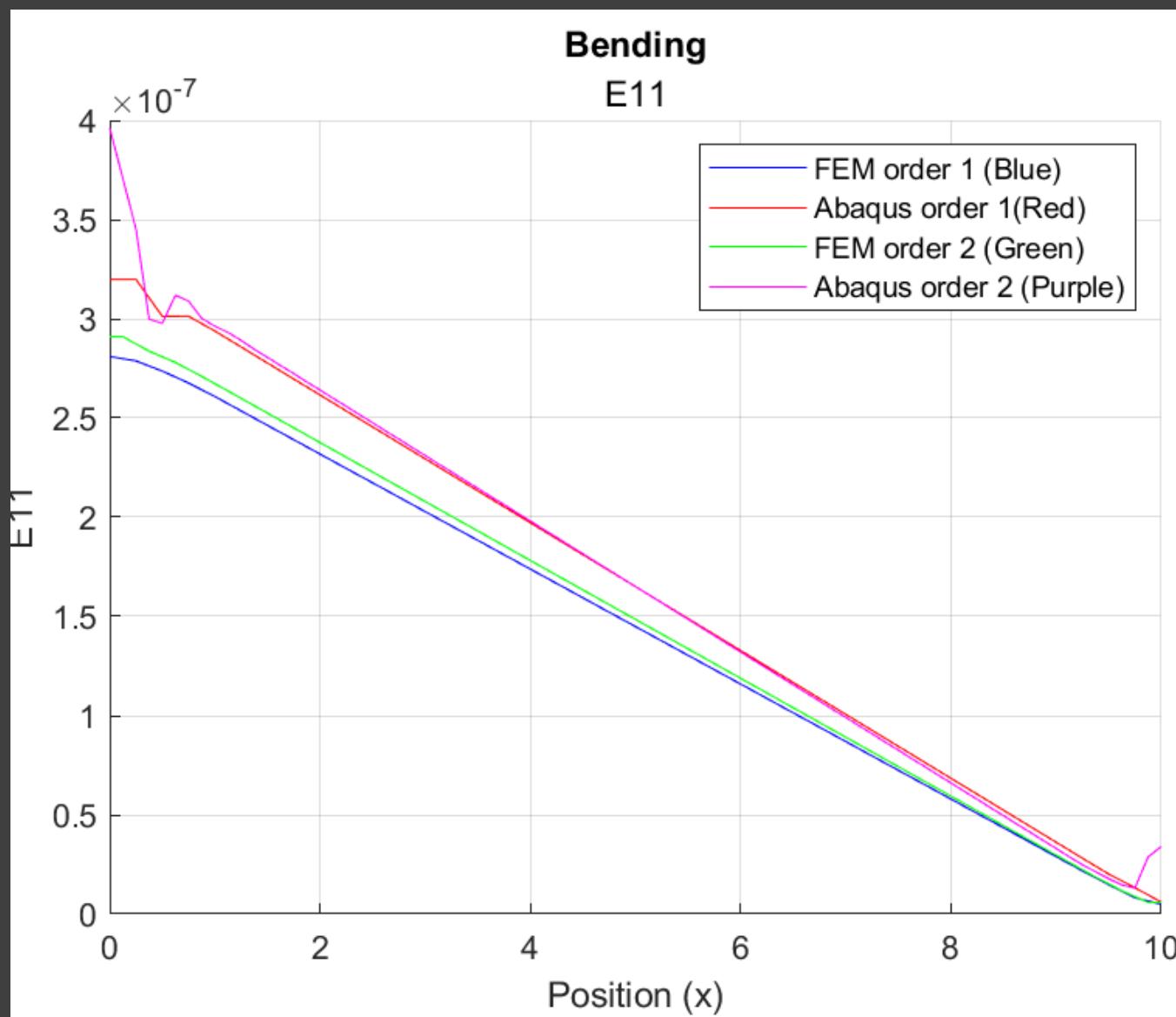
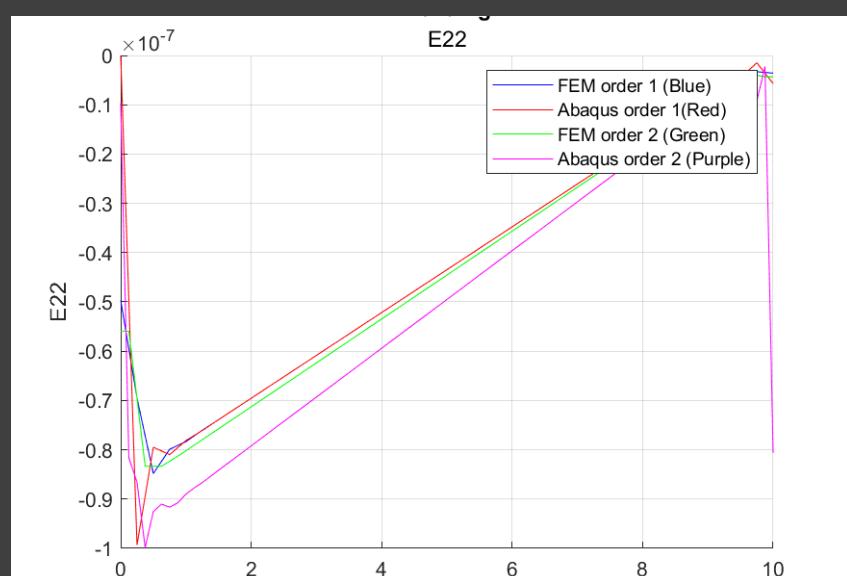
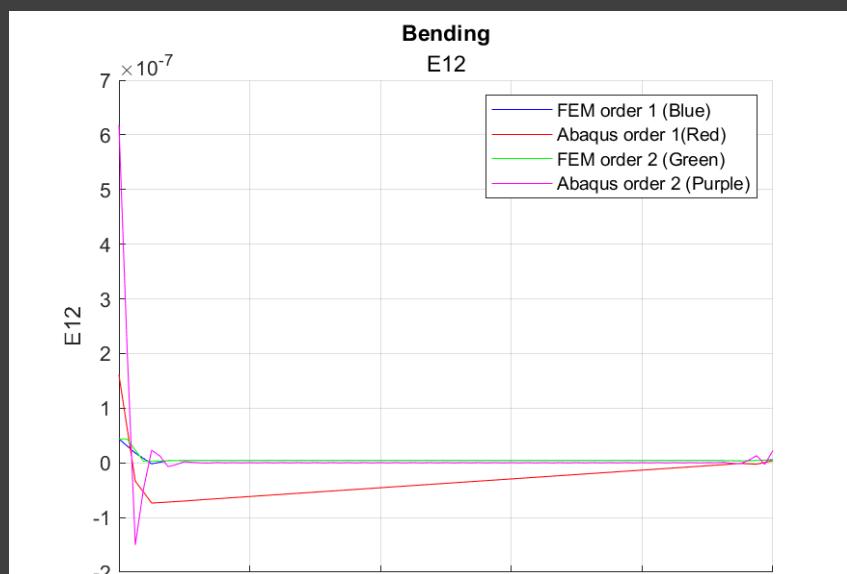
```
                med(index, 1) = med(index, 1) + 1;
```

```
                inode = inode + 1;
```

RESULTADOS

DEFORMAÇÃO





```
case 2
    for iel = 1:size(obj.Matrix.Pre.Mesh.Connectivity, 1) % for each element
        inode = 1; % for each node in the element
        for iwy = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in y
            for iwx = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in x
                for icomp = 1:(obj.Matrix.Pre.Mesh.Dim+(obj.Matrix.Pre.Mesh.Dim^2-obj.Matrix.Pre.Mesh.Dim)/2) % for each comp
```

```
                index = sort(obj.Matrix.Pre.Mesh.Connectivity(iel, 2:end));
```

```
                e(index, icomp) = e(index, icomp) + obj.E(iel, iwx, iwy, icomp);
```

```
                s(index, icomp) = s(index, icomp) + obj.S(iel, iwx, iwy, icomp);
```

```
            end
```

```
            mis(index, 1) = mis(index, 1) + obj.Mises(iel, iwx, iwy);
```

```
            med(index, 1) = med(index, 1) + 1;
```

```
            inode = inode + 1;
```

```
        end
```

```
    end
```

```
end
```

```
case 3
```

```
for iel = 1:size(obj.Matrix.Pre.Mesh.Connectivity, 1) % for each element
```

```
    inode = 1; % for each node in the element
```

```
    for iwj = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in z
```

```
        for iwy = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in y
```

```
            for iwx = 1:obj.Matrix.Pre.Mesh.Order+1 % for each integration point in x
```

```
                for icomp = 1:(obj.Matrix.Pre.Mesh.Dim+(obj.Matrix.Pre.Mesh.Dim^2-obj.Matrix.Pre.Mesh.Dim)/2) % for each comp
```

```
                    index = sort(obj.Matrix.Pre.Mesh.Connectivity(iel, 2:end));
```

```
                    e(index, icomp) = e(index, icomp) + obj.E(iel, iwx, iwy, iwj, icomp);
```

```
                    s(index, icomp) = s(index, icomp) + obj.S(iel, iwx, iwy, iwj, icomp);
```

```
                end
```

```
                mis(index, 1) = mis(index, 1) + obj.Mises(iel, iwx, iwy, iwj);
```

```
                med(index, 1) = med(index, 1) + 1;
```

```
                inode = inode + 1;
```

RESULTADOS

TENSÃO



