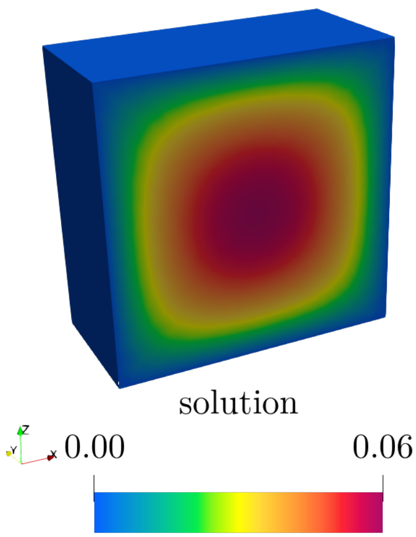
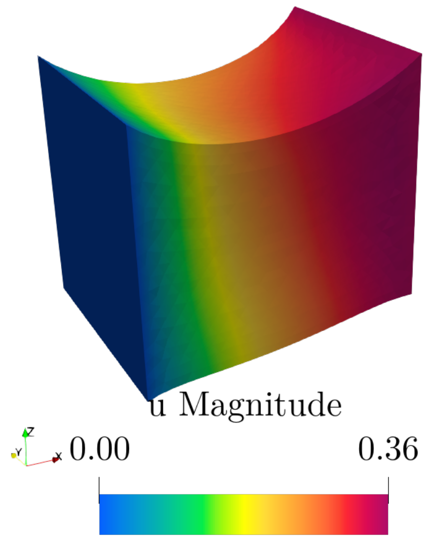


(a)



(b)

Figure 1: (a) Solution to exercise 1. The domain was clipped along the plane $y = 0.5$. (b) Solution to exercise 2. The domain was warped by the solution $\mathbf{u}$ (using the filter "Warp by vector").

Exercise 2.

Let $\Omega = (0,1)^3$ be the unit cube and let us consider the following linear elasticity problem: find a displacement field $\mathbf{u} : \Omega \to \mathbb{R}^3$ such that

$$\begin{cases} -\boldsymbol{\nabla} \cdot \sigma(\mathbf{u}) = \mathbf{f} & \text{in } \Omega \ , \\ \mathbf{u} = \mathbf{g} & \text{on } \Gamma_0 \cup \Gamma_1 \ , \\ \sigma(\mathbf{u})\mathbf{n} = \mathbf{0} & \text{on } \Gamma_2 \cup \Gamma_3 \cup \Gamma_4 \cup \Gamma_5 \ , \end{cases} \tag{2}$$

where

$$\begin{aligned} \sigma(\mathbf{u}) &= \mu \boldsymbol{\nabla} \mathbf{u} + \lambda (\boldsymbol{\nabla} \cdot \mathbf{u}) I \ , \\ \Gamma_0 &= \{x = 0, y \in (0,1), z \in (0,1)\} \ , \\ \Gamma_1 &= \{x = 1, y \in (0,1), z \in (0,1)\} \ , \\ \Gamma_2 &= \{x \in (0,1), y = 0, z \in (0,1)\} \ , \\ \Gamma_3 &= \{x \in (0,1), y = 1, z \in (0,1)\} \ , \\ \Gamma_4 &= \{x \in (0,1), y \in (0,1), z = 0\} \ , \\ \Gamma_5 &= \{x \in (0,1), y \in (0,1), z = 1\} \ , \end{aligned}$$

$\mu = 1$, $\lambda = 10$, $\mathbf{g}(\mathbf{x}) = (0.25x, 0.25x, 0)^T$ and $\mathbf{f}(\mathbf{x}) = (0, 0, -1)^T$.

2.1. Write the weak formulation of problem (2).

Solution. Since the unknown of the problem is vector-valued, we work in spaces of vector-valued functions (denoted in bold). Other than that, the procedure for deriving a weak formulation is essentially unchanged, provided we use the appropriate notions of product (e.g. scalar product between vectors or tensors) and spatial derivatives.

Let us introduce the function spaces

$$
\begin{aligned}
V_0 &= \left\{ \mathbf{v} \in \left[H^1(\Omega)\right]^3 \ : \ \mathbf{v} = \mathbf{0} \text{ on } \Gamma_0 \cup \Gamma_1 \right\} \ , \\
V &= \left\{ \mathbf{v} \in \left[H^1(\Omega)\right]^3 \ : \ \mathbf{v} = \mathbf{g} \text{ on } \Gamma_0 \cup \Gamma_1 \right\} \ .
\end{aligned}
$$

Let $\mathbf{v} \in V_0$, we get:

$$
\int_\Omega \left(\mu \boldsymbol{\nabla} \mathbf{u} \colon \boldsymbol{\nabla} \mathbf{v} + \lambda (\boldsymbol{\nabla} \cdot \mathbf{u}) I \colon \boldsymbol{\nabla} \mathbf{v} \right) d\mathbf{x} = \int_\Omega \mathbf{f} \cdot \mathbf{v} d\mathbf{x} \ .
$$

Introducing

$$
\begin{aligned}
a(\mathbf{u}, \mathbf{v}) &= \int_\Omega \left(\mu \boldsymbol{\nabla} \mathbf{u} \colon \boldsymbol{\nabla} \mathbf{v} + \lambda (\boldsymbol{\nabla} \cdot \mathbf{u}) (\boldsymbol{\nabla} \cdot \mathbf{v}) \right) d\mathbf{x} \ , \\
F(\mathbf{v}) &= \int_\Omega \mathbf{f} \cdot \mathbf{v} d\mathbf{x} \ ,
\end{aligned}
$$

the weak formulation reads:

$$
\text{find } \mathbf{u} \in V \text{ such that } a(\mathbf{u}, \mathbf{v}) = F(\mathbf{v}) \text{ for all } \mathbf{v} \in V_0 \ .
$$

We can equivalently formulate the problem in a setting where the Lax-Milgram lemma is applicable by introducing a lifting function $\mathbf{R}(g) \in V$ such that $\mathbf{R}(\mathbf{g}) = \mathbf{g}$ on $\Gamma_0 \cup \Gamma_1$. Then, setting $\mathbf{u} = \mathbf{u}_0 + \mathbf{R}(\mathbf{g})$, the problem reads:

$$
\text{find } \mathbf{u}_0 \in V_0 \text{ such that } a(\mathbf{u}_0, \mathbf{v}) = F(\mathbf{v}) - a(\mathbf{R}(\mathbf{g}), v) \text{ for all } \mathbf{v} \in V_0 \ .
$$

2.2. Implement in `deal.II` a finite element solver for problem (2).

Solution. See the file `src/lab-07-exercise2.cpp`. The solution is displayed in Figure 1b.

When introducing the discrete weak formulation to (2), it is convenient to work with vector-valued basis functions. In other words, we assume the discrete solution can be decomposed as

$$
\mathbf{u}_h(\mathbf{x}) = \sum_{i=1}^{N_h} U_i \boldsymbol{\varphi}_i(\mathbf{x}) \ ,
$$

where $\boldsymbol{\varphi}_i$ are the vector-valued basis functions. There are different ways in which they can be constructed. One possible way, starting from the scalar basis functions φ_j (with $j = 1, 2, \ldots, N_h/3$), is the following:

$$\boldsymbol{\varphi}_1 = \begin{bmatrix} \varphi_1 \\ 0 \\ 0 \end{bmatrix} \;, \quad \boldsymbol{\varphi}_2 = \begin{bmatrix} 0 \\ \varphi_1 \\ 0 \end{bmatrix} \;, \quad \boldsymbol{\varphi}_3 = \begin{bmatrix} 0 \\ 0 \\ \varphi_1 \end{bmatrix} \;, \quad \boldsymbol{\varphi}_4 = \begin{bmatrix} \varphi_2 \\ 0 \\ 0 \end{bmatrix} \;, \quad \ldots$$

This kind of choice (i.e. each vector basis function has only one non-zero component, and its value for that component is that of a scalar basis function) is referred to as *primitive* in `deal.II`. Of course, the ordering is arbitrary (for instance, we could first have all basis functions associated to the first component, then all those associated to the second, then all those associated to the third).

`deal.II` allows to access vector-valued basis functions in two ways:

1. for a given index i of a vector-valued basis function, we can access the index c of its non-zero component through `FiniteElement::system_to_component_index` and the valueof the associated scalar basis function through `FEValues::shape_value` (and similar for e.g. the gradient or other derivatives);

2. combining `FEValues` and `FEValuesExtractors` to obtain `FEValuesView`, we can access the vector-valued shape function directly.

Generally, the second approach is more convenient to program, because it leads to code that more closely matches the mathematical notation. For this reason, this is the approach followed in the solution. The first approach may be marginally more efficient in some circumstances (especially as it is less prone to causing the construction of temporary `Tensor`s), but this should be verified through a profiler.