`PytHHOn3D` documentation

David Siedel Olivier Fandeur Thomas Helfer

01/08/2020

1 Introduction

The `PytHHOn3D` library is a PDE solver implementing the HHO method in a generic and object-oriented fashion using `python`, with specific applications to solid mechanics : the structure of the code follows that of a usual FEM solver, namely :

- `PytHHOn3D` reads and parses a mesh file as input
- it takes a set of input data to set the PDE problem to solve (boundary conditions, volumetric forces, HHO elements characteristics)
- `PytHHOn3D` builds the framework specific to HHO methods, and the HHO element for each element in the mesh
- it solves a linear system given a set of fourth order tensors (one for each element) as tangent matrices
- it returns the unknown values at vertices and quadrature points

2 Model problem : small strain linear elasticity

Problem setting Let Ω a domain in $\mathbb{R}^d$ with Lipschitz boundaries $\partial\Omega$. Let the following regular functional spaces :

Definition : **Sobolev space**

$$H^1(\Omega;\mathbb{R}^d) = \left\{\underline{\boldsymbol{u}} \in L^2(\Omega;\mathbb{R}^d), \nabla\underline{\boldsymbol{u}} \in L^2(\Omega;\mathbb{R}^{d\times d})\right\}$$

Definition : **Divergence Sobolev space**

$$H^1_{div}(\Omega;\mathbb{R}^{d\times d}) = \left\{\underset{\sim}{\boldsymbol{\tau}} \in L^2(\Omega;\mathbb{R}^{d\times d}), \nabla\cdot\underset{\sim}{\boldsymbol{\tau}} \in L^2(\Omega;\mathbb{R}^d)\right\}$$

Problem Let the small strain linear elastic problem for $\underline{\boldsymbol{u}}$ such that:

$$\begin{cases} \nabla\cdot\underset{\sim}{\boldsymbol{\sigma}} = -\underline{\boldsymbol{f}} & \text{in } \Omega \\ \underset{\sim}{\boldsymbol{\sigma}} = \underline{\boldsymbol{t}}\cdot\underline{\boldsymbol{n}} & \text{on } \partial\Omega_N \\ \underline{\boldsymbol{u}} = \underline{\boldsymbol{u}}_D & \text{on } \partial\Omega_D \\ \underset{\sim}{\boldsymbol{\sigma}} = 2\mu\underset{\sim}{\boldsymbol{\varepsilon}} + \lambda\,\mathrm{Tr}(\underset{\sim}{\boldsymbol{\varepsilon}}) = \underset{\approx}{\mathbb{C}} : \underset{\sim}{\boldsymbol{\varepsilon}} & \text{in } \Omega \\ \underset{\sim}{\boldsymbol{\varepsilon}} = \frac{1}{2}(\nabla\underline{\boldsymbol{u}} + \nabla^T\underline{\boldsymbol{u}}) = \nabla^s\underline{\boldsymbol{u}} & \text{in } \Omega \end{cases} \tag{1}$$

Where

- $\underset{\sim}{\boldsymbol{\sigma}} \in H^1_{div}(\Omega;\mathbb{R}^{d\times d})$ is the symmetric Cauchy stress tensor
- $\underline{\boldsymbol{u}} \in H^1(\Omega;\mathbb{R}^d)$ is the displacement field
- $\underline{\boldsymbol{t}} \in H^1(\Omega;\mathbb{R}^d)$ are the applied forces
- $\underline{\boldsymbol{u}}_D \in H^1(\Omega;\mathbb{R}^d)$ is the imposed displacement
- λ and μ are Lamé coefficients of the material

Weak form The weak form of (1) reads as the Principle of Virtual Works :

Theorem : **Principle of Virtual Works (PVW)**

find $\underline{\boldsymbol{u}} \in H^1(\Omega;\mathbb{R}^d)$ such that :

$$\begin{cases} \displaystyle\int_\Omega \underset{\sim}{\boldsymbol{\sigma}} : \nabla^s\underline{\boldsymbol{v}} = \int_\Omega \underline{\boldsymbol{f}}\cdot\underline{\boldsymbol{v}} + \int_{\partial\Omega_N} \underline{\boldsymbol{t}}\cdot\underline{\boldsymbol{v}} & \forall\underline{\boldsymbol{v}} \in H^1(\Omega;\mathbb{R}^d) \\ \underline{\boldsymbol{u}} = \underline{\boldsymbol{u}}_D & \text{on } \partial\Omega_D \end{cases} \tag{2}$$

Discretization : the Finite Element Method (FEM) The FEM method consists in descretizing (2) in both physical and functional spaces through Lagrange polynbomials bases : defining the usual shape functions, that are polynomials of a given order k_c, such that they are of value 1 at a given node and 0 elsewhere in Ω, one can put (2) in a matricial form, which enables to find an approximation of $\underline{\boldsymbol{u}}$ in (2) numerically.

Polynomial approximation Let $\mathbb{P}^{k_c}(\Omega; \mathbb{R}^d) \subset H^1(\Omega; \mathbb{R}^d)$ a polynomial space of order k_c. Let N_k^d the dimension of $\mathbb{P}^{k_c}(\Omega; \mathbb{R}^d)$, and $(\underline{\boldsymbol{\phi}}_m)_{m \leq N_{k_c}^d}$ a basis of $\mathbb{P}^{k_c}(\Omega; \mathbb{R}^d)$. N_k^d is then the number of vectors $\underline{\boldsymbol{\phi}}_m$ composing

3 The HHO method

Polynomial basis Let the scaled polynomial basis

Definition : **Scaled monomial exponents**

Let $k_j \geq 0$ and $d \geq 1$ two integers.

$$\alpha(k_j, d) = \left\{ \underline{\boldsymbol{\alpha}} = (\alpha_1, ..., \alpha_d) \;\middle|\; \sum_{1 \leq i \leq d} \alpha_i = k_j \right\} \tag{3}$$

For a given integer k, we define :

$$\alpha_{mono}(k, d) = \left\{ \alpha(k_j, d) \;\middle|\; 0 \leq k_j \leq k \right\} \tag{4}$$

Remark : **Scaled monomial exponents**
Let $k_j \geq 0$ and $d \geq 1$ two integers.

$$\alpha(k_j, d) = \left\{ \underline{\boldsymbol{\alpha}} = (\alpha_1, ..., \alpha_d) \;\middle|\; \sum_{1 \leq i \leq d} \alpha_i = k_j \right\} \tag{5}$$

For a given integer k, we define :

$$\alpha_{mono}(k, d) = \left\{ \alpha(k_j, d) \;\middle|\; 0 \leq k_j \leq k \right\} \tag{6}$$

Definition : **Scaled monomial basis**

Let $D \subset \mathbb{R}^d$ a closed domain of volume $\boldsymbol{v}_D$ and of barycenter $\underline{\boldsymbol{x}}_D$. The (sclaed) monomial basis of polynbomials $\mathcal{B}_{mono}(k,d)$ of order k in D writes as :

$$\mathcal{B}_{mono}(k,d) = \left\{ \prod_{\boldsymbol{\alpha} \in \underline{\boldsymbol{\alpha}}} \left(\frac{\underline{\boldsymbol{x}} - \underline{\boldsymbol{x}}_D}{\boldsymbol{v}_D} \right)^{\boldsymbol{\alpha}} \;\middle|\; \underline{\boldsymbol{\alpha}} \in \mathcal{A}_{mono}(k,d) \right\} \tag{7}$$

In particular, the dimension N_k^d of $\mathcal{B}_{mono}(k,d)$ is $N_d^k = \binom{d+k}{k}$

References