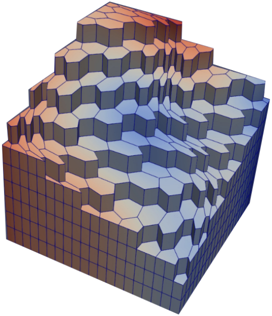
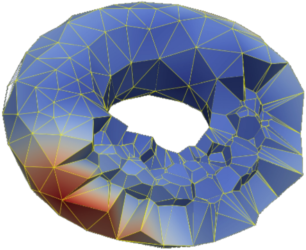
The ParaSkel library, a toolbox for the polytopal approximation of PDEs



Simon Lemaire
(INRIA, Univ. Lille)



Gmsh/NEMESIS Workshop, Montpellier
January 26, 2026

Why polytopal elements?

Why polytopal elements?

- for the very same reasons the Incas were already using them!

Figure: The twelve-angled stone (Cusco, Peru), XIIIth century approximately.

The skeletal family (1/3)

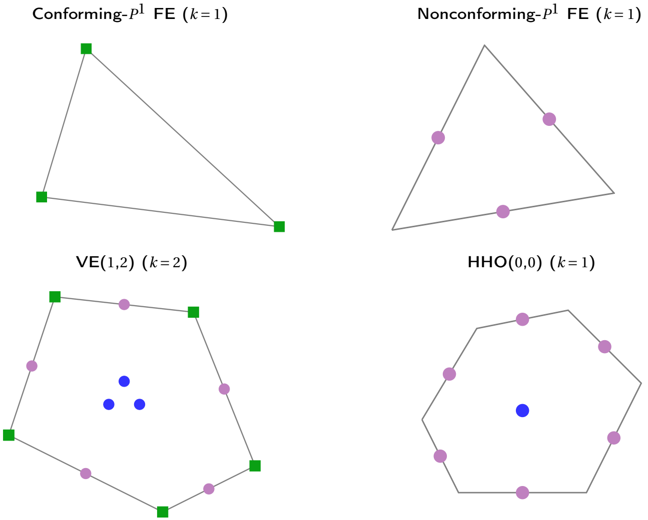


Figure: Examples of H^1 skeletal elements of order $k \in \mathbb{N}^\star$ (i.e., P^k-exact) in 2D.

The skeletal family (2/3)

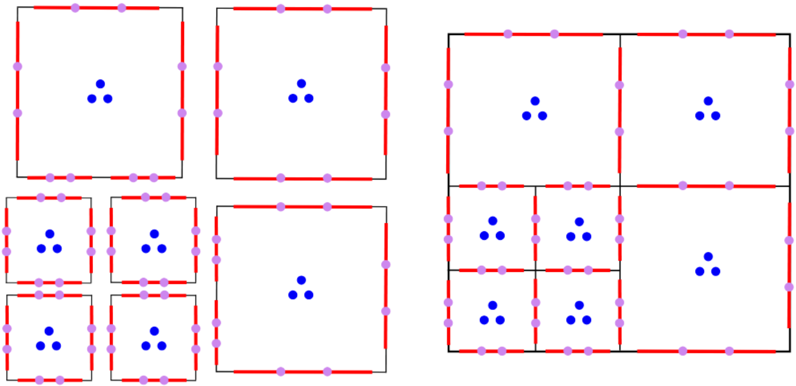


Figure: Assembly procedure for skeletal methods (example of HHO(1,1) ($k = 2$) in 2D).[1]

[1] illustration borrowed from [Cicuttin, Ern, Pignet; 21]

The skeletal family (3/3)

Two types of DoFs

- those attached to the mesh skeleton: tagged by ∂
- those attached to the mesh bulk (if any): tagged by $\circ$

Common algebraic structure

Skeletal methods yield linear systems of the form

$$\mathbb{A}_{\mathcal{D}} \boldsymbol{U}_{\mathcal{D}} = \boldsymbol{F}_{\mathcal{D}},$$

whith matrices $\mathbb{A}_{\mathcal{D}}$ such that

$$\boldsymbol{V}_{\mathcal{D}}^{\top} \mathbb{A}_{\mathcal{D}} \boldsymbol{W}_{\mathcal{D}} = \sum_{K \in \mathcal{K}} \boldsymbol{V}_K^{\top} \mathbb{A}_K \boldsymbol{W}_K \qquad \forall (\boldsymbol{V}_{\mathcal{D}}, \boldsymbol{W}_{\mathcal{D}}),$$

where $\boldsymbol{V}_K := \boldsymbol{V}_{\mathcal{D}|K} = (\boldsymbol{V}_K^{\circ}, \boldsymbol{V}_K^{\partial})$ and $\boldsymbol{W}_K := \boldsymbol{W}_{\mathcal{D}|K} = (\boldsymbol{W}_K^{\circ}, \boldsymbol{W}_K^{\partial})$.

$\rightsquigarrow$ as a by-product, one may write $\mathbb{A}_{\mathcal{D}} = \begin{pmatrix} \mathbb{A}_{\mathcal{D}}^{\circ\circ} & \mathbb{A}_{\mathcal{D}}^{\circ\partial} \\ \mathbb{A}_{\mathcal{D}}^{\partial\circ} & \mathbb{A}_{\mathcal{D}}^{\partial\partial} \end{pmatrix}$ with $\mathbb{A}_{\mathcal{D}}^{\circ\circ} = \mathrm{diag}\big((\mathbb{A}_K^{\circ\circ})_{K \in \mathcal{K}}\big)$

Factorizable implementation

- no reference element: all computations are performed over the physical element
- local elimination of $\circ$ DoFs: $\mathbb{A}_K^{\mathrm{S}} := \mathbb{A}_K^{\partial\partial} - \mathbb{A}_K^{\partial\circ} [\mathbb{A}_K^{\circ\circ}]^{-1} \mathbb{A}_K^{\circ\partial}$ for all $K \in \mathcal{K}$
- global numbering of ∂ DoFs: exhaustive enumeration of possible ∂ DoF locations
- global assembly of the condensed linear system: $\mathbb{A}_{\mathcal{D}}^{\mathrm{S}}$ built from the $(\mathbb{A}_K^{\mathrm{S}})_{K \in \mathcal{K}}$

Skeletal vs. DG

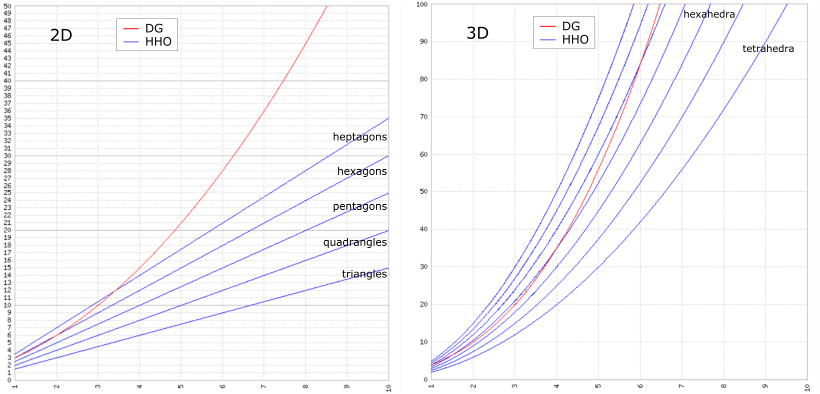


Figure: #DoFs/cell vs. k for (condensed) HHO and IP-DG on a Poisson problem with Dirichlet BCs.

The (H^1) skeletal zoo on polytopes

Lowest-order ancestors

- fully conforming methods:
  - ⇝ (nodal) Mimetic Finite Difference (MFD) method [Brezzi, Buffa, Lipnikov; 09]
  - ⇝ Vertex Approximate Gradient (VAG) scheme [Eymard, Guichard, Herbin; 12]
  - ⇝ (vertex-based) Compatible Discrete Operator (CDO) scheme [Bonelle, Ern; 14]
- weakly conforming methods:
  - ⇝ Hybrid Finite Volume (HFV) method [Eymard, Gallouët, Herbin; 10]
  - ⇝ (face-based) Compatible Discrete Operator (CDO) scheme [Bonelle, Ern; 14]
  - ⇝ generalized Crouzeix–Raviart method [Di Pietro, Lemaire; 15]

Arbitrary-order successors

- fully conforming methods:
  - ⇝ conforming Virtual Element (cVE) method [Beirão da Veiga, Brezzi, Cangiani, Manzini, Marini, Russo; 13]
  - ⇝ Discrete De Rham (DDR) method [Di Pietro, Droniou, Rapetti; 20]
- weakly conforming methods:
  - ⇝ Hybridizable Discontinuous Galerkin (HDG) method [Cockburn, Gopalakrishnan, Lazarov; 09], [Lehrenfeld; 10], [Lehrenfeld, Schöberl; 16], [Oikawa; 15]
  - ⇝ Weak Galerkin (WG) method [Wang, Ye; 13]
  - ⇝ Hybrid High-Order (HHO) method [Di Pietro, Ern, Lemaire; 14]
  - ⇝ nonconforming Virtual Element (ncVE) method [Lipnikov, Manzini; 14], [Ayuso de Dios, Lipnikov, Manzini; 16]

The ParaSkel library (1/3)

Main features

Eventually, the ParaSkel library is expected to possess 5 main assets:

- a unified 2D/3D implementation;
- the native support of any type of DoFs (vertex-, edge-, face-, and cell-based);
- a factorized architecture (with common-to-all-methods local elimination and global assembly);
- the use of efficient quadrature formulas on polytopes (without the need for subtessellation);
- the embedding of parallel computation capabilities.

In practice

- programming languages: C++, MPI/OpenMP
- Open license: GNU LGPL v3
- GitLab repository: `https://gitlab.inria.fr/simlemai/paraskel`
- how to cite: hal-03517921 (SWH deposit)

Development team

- Simon Lemaire: instigator and coordinator (since 2019)
- Laurence Beaude: lead developer (from 02/2020 to 08/2021)
- Thoma Zoto: lead developer (from 12/2022 to 06/2024)
- other contributors. . .

Polytopal quadrature

- aim: avoid the subtessellation of the element [Chin, Lasserre, Sukumar; 15]
- numerical integration via Stokes formula: for $\psi \in \mathbb{H}^p$, $\psi = \mathrm{div}(\boldsymbol{\Psi})$ with $\boldsymbol{\Psi}(\boldsymbol{x}) = \frac{\boldsymbol{x}\psi(\boldsymbol{x})}{d+p}$, thus

$$\int_K \psi = \int_{\partial K} \boldsymbol{\Psi} \cdot \boldsymbol{n}$$

Hybrid parallelism

- local computations (in each $K \in \mathcal{K}$) are embarrassingly parallel
- supported (CPU) architectures: distributed (MPI) and/or shared (OpenMP) memory
- parallel linear solvers: interfacing of PETSc

Polytopal mesher

- support of different mesh formats (among which Gmsh)
- Voronoï mesher: interfacing of VoroCrust (Sandia National Labs)

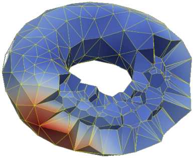


Figure: Voronoï tessellation of a torus.

Quick example: HHO for Stokes

```
#define DIM 3     // the spatial dimension must be known at compilation time

// Initialize the HHO<DIM, H1d, L2> discrete space
MyDiscreteSpace::DiscreteSpace<DIM> ds(mesh.get());
ds.init<MyDiscreteSpace::HHO, MyDiscreteSpace::H1d, MyDiscreteSpace::L2>(mesh.get(), bulk_k, skeletal_k);

// Add velocity stiffness
ds.add_stiffness_contribution(var_u, mu);
// Add velocity-pressure coupling
ds.add_divergence_coupling(var_u, var_p, -1.);
// Add a Lagrange multiplier for the pressure
ds.add_lagrange_multiplier(var_p);

// Set Dirichlet (essential) BC for the velocity
ds.set_BCtype(mesh.get(), var_u, MyDiscreteSpace::ESSENTIAL);

// Add loading to rhs
ds.add_loading_to_rhs(continuous_rhs, specific_quadra_order, var_u);

// Initialize matrix pattern and assemble local contributions into global system
MySolver::MatrixPattern<DIM> mp(mesh.get(), &ds, static_condensation, essentialBC_elimination);
mp.assemble_local_contributions(mesh.get(), &ds);
```

ANY QUESTIONS?