ngsVEM: a lightning Virtual Element Method for NGSolve

Umberto Zerbinati[1]

[1] *Mathematical Institute, University of Oxford*

NGSolve Users' Meeting, 2026

UNIVERSITY OF OXFORD
Mathematical Institute

Oxford Mathematics

THE STORY IN ONE SLIDE

What is ngsVEM?

A general-order **lightning Virtual Element Method** as a custom NGSolve finite-element space, on **polygonal meshes** built by agglomerating Netgen triangles.

THE STORY IN ONE SLIDE

Mathematical Institute

What is ngsVEM?

A general-order **lightning Virtual Element Method** as a custom NGSolve finite-element space, on **polygonal meshes** built by agglomerating Netgen triangles.

The lightning idea

Each polygon basis function is approximated *explicitly* by a rational function with poles clustered outside the corners. The local stiffness is then just $\int_K \nabla\varphi_i \cdot \nabla\varphi_j$ – **no stabilization, no projection operators**.

THE STORY IN ONE SLIDE

What is ngsVEM?

A general-order **lightning Virtual Element Method** as a custom NGSolve finite-element space, on **polygonal meshes** built by agglomerating Netgen triangles.

The lightning idea

Each polygon basis function is approximated *explicitly* by a rational function with poles clustered outside the corners. The local stiffness is then just $\int_K \nabla\varphi_i \cdot \nabla\varphi_j$ – **no stabilization, no projection operators**.

Pointwise access to the discrete solution makes the lightning VEM a natural **coarse engine** for multigrid – here for Poisson and high-order elasticity.

VIRTUAL ELEMENT SPACES

On each polygon K the VEM chooses the local space

$$V_h^k(K) := \left\{ v_h \in H^1(K) \ : \ \Delta v_h \in \mathbb{P}_{k-2}(K),\ v_h|_{\partial K} \in \mathbb{B}_k(\partial K) \right\},$$

with $\mathbb{B}_k(\partial K)$ the continuous, edge-wise polynomials of degree k.

- At lowest order $k = 1$: functions are **harmonic** inside K, **piecewise linear** on ∂K, the degrees of freedom are the **vertex values**.
- These functions are *almost never* known in closed form.

VIRTUAL ELEMENT SPACES

On each polygon K the VEM chooses the local space

$$V_h^k(K) := \left\{ v_h \in H^1(K) \ : \ \Delta v_h \in \mathbb{P}_{k-2}(K),\ v_h|_{\partial K} \in \mathbb{B}_k(\partial K) \right\},$$

with $\mathbb{B}_k(\partial K)$ the continuous, edge-wise polynomials of degree k.

- At lowest order $k = 1$: functions are **harmonic** inside K, **piecewise linear** on ∂K, the degrees of freedom are the **vertex values**.
- These functions are *almost never* known in closed form.

> Classical VEM is a **Galerkin method with** a polynomial-consistent part *plus an ad-hoc stabilization term*. We replace the unknown basis by an explicit one instead.

THE LIGHTNING APPROXIMATION

Approximate each basis function *explicitly* by a rational function

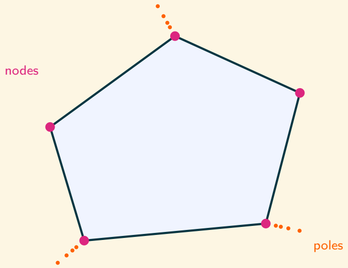
$$\hat{\varphi}_i = \text{Real part}\left\{ \sum_j \frac{a_j}{z - z_j} + \sum_j b_j\, z^j \right\}, \quad z = x + iy,$$

with poles z_j clustered **exponentially** toward the polygon corners (the lightning Laplace solver).

The basis is known *pointwise*; thus, the local stiffness is simply

$$a^K(\hat{\varphi}_i, \hat{\varphi}_j) = \int_K \nabla \hat{\varphi}_i \cdot \nabla \hat{\varphi}_j \, dx.$$

Classical VEM never sees $\hat{\varphi}_i$, so it must replace it by a polynomial **projection** $\Pi\hat{\varphi}_i$ plus stabilization. Here there is **no projection**.



Poles (●) cluster *near* the corners; nodal degrees of freedom (●) are the corner values.

WHY LIGHTNING VEM?

Compared with classical VEM, the lightning VEM:

1. needs **no stabilization term**;
2. needs **no projection operators –** implemented like an ordinary "non-reference-element" finite element;
3. has **better-conditioned** stiffness matrices as elements distort;
4. gives **pointwise access** to the discrete solution.

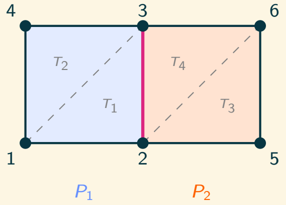
IMPLEMENTATION IN NGSOLVE

- Polygons are built by **agglomerating Netgen triangles** (`polysize` triangles per polygon, `polysize=1` recovers ordinary P1 triangles).

  The agglomerated triangles still triangulate each polygon, so standard NGSolve **element-by-element assembly** is reused.
- Each sub-triangle evaluates its parent polygon's lightning basis at the mapped point, and $\int_T \nabla\varphi_i \cdot \nabla\varphi_j$ is summed over the polygon's triangles.
- Shared corner degrees of freedom and linear edge traces give H^1-conformity across edges.

```
from ngsVEM import LightningVEMSpace
fes = LightningVEMSpace(mesh, order=1, polysize=4, dirichlet=".*")
u, v = fes.TnT();  a = BilinearForm(grad(u)*grad(v)*dx).Assemble()
```

HOW THE MATRIX IS ASSEMBLED



$P_1 = \{1,2,3,4\}$, $P_2 = \{2,3,5,6\}$.

Shared edge $2-3$.

Each polygon carries **4** nodal basis functions $\varphi_1^P, \ldots, \varphi_4^P$ (one per corner). Its local stiffness is a 4×4 matrix, built by **summing over the polygon's two sub-triangles**:

$$A_{ij}^{P_1} = \int_{T_1} \nabla\varphi_i^{P_1} \cdot \nabla\varphi_j^{P_1} + \int_{T_2} \nabla\varphi_i^{P_1} \cdot \nabla\varphi_j^{P_1},$$

$$A_{ij}^{P_2} = \int_{T_3} \nabla\varphi_i^{P_2} \cdot \nabla\varphi_j^{P_2} + \int_{T_4} \nabla\varphi_i^{P_2} \cdot \nabla\varphi_j^{P_2}.$$

The same lightning basis φ^P is evaluated on *each* sub-triangle, exactly NGSolve's ordinary element loop, with a polygon in place of a triangle.

HOW THE MATRIX IS ASSEMBLED

The two 4×4 blocks are **scattered** into the global 6×6 matrix at their polygon's corner indices. The **shared** corners $2, 3$ receive a contribution from *both* polygons – they are **added**:

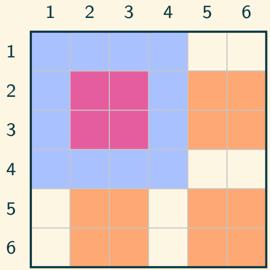
$$\boxed{A_{22} = A_{22}^{P_1} + A_{22}^{P_2}}, \qquad A_{23} = A_{23}^{P_1} + A_{23}^{P_2}.$$

entries among $\{1, 2, 3, 4\}$ only: from P_1 ■;

entries among $\{2, 3, 5, 6\}$ only: from P_2 ■;

entries among $\{2, 3\}$: **summed** ■;

e.g. $(1, 5)$: *zero* – vertices share no polygon.



global A (6×6)

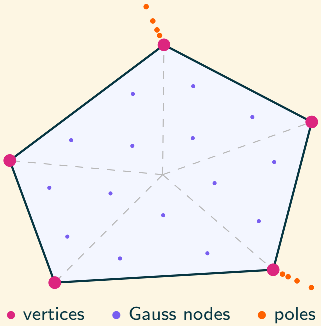
Central 2×2 block (■) is the summed, shared part.

QUADRATURE: INTEGRATING THE RATIONAL BASIS

Each stiffness entry splits over the polygon's sub-triangles:

$$A_{ij}^K = \int_K \nabla\hat{\varphi}_i \cdot \nabla\hat{\varphi}_j = \sum_{T\subset K} \int_T \nabla\hat{\varphi}_i \cdot \nabla\hat{\varphi}_j.$$

- The agglomerated Netgen triangles **already triangulate** K – reuse them as the integration mesh and NGSolve's **element-wise Gauss loop**.
- $\hat{\varphi}$ is the explicit rational function, so $\hat{\varphi}$ and $\nabla\hat{\varphi}$ are evaluated **in closed form** at every quadrature node – no reference map, no projection.
- Poles sit *just outside* the corners: the integrand is smooth inside K but **steep near corners**, so a **high-order** rule is used.



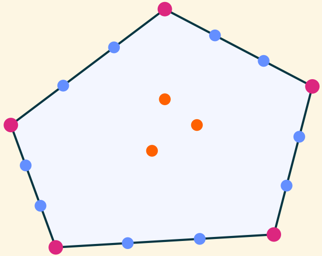
Gauss points live on the sub-triangles; poles cluster *just outside* the corners.

Exact, cheap pointwise evaluation ⇒ the quadrature order is raised until each $\int_T$ is converged. Quadrature is the *only* consistency error – no stabilization, no projection.

HIGH-ORDER SPACES ($k \geq 1$)

On a polygon E with m vertices the order-k space has $N_E = mk + k(k-1)/2$ degrees of freedom:

- **vertex** values at the m corners;
- **edge** values at $k-1$ Gauss–Lobatto nodes per edge;
- **interior** moments $\mu_\alpha(v) = \frac{1}{|E|}\int_E v\,((x - x_E)/h_E)^\alpha$, $|\alpha| \leq k-2$.



$k = 3$ on a pentagon:
• vertex • edge • interior.

$\mathbb{P}_k \subset \hat{V}_h^k$ reproduced to machine precision.

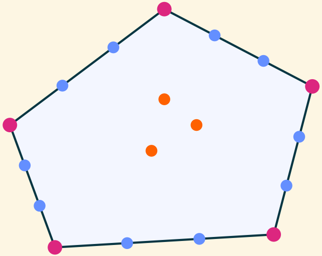
HIGH-ORDER SPACES ($k \geq 1$)

On a polygon E with m vertices the order-k space has $N_E = mk + k(k-1)/2$ degrees of freedom:

- **vertex** values at the m corners;
- **edge** values at $k-1$ Gauss–Lobatto nodes per edge;
- **interior** moments $\mu_\alpha(v) = \frac{1}{|E|}\int_E v\,((x - x_E)/h_E)^\alpha$, $|\alpha| \leq k-2$.

The basis comes from two lightning families:

1. mk **harmonic** solves with degree-k boundary traces;
2. $k(k-1)/2$ **interior** functions $\varphi = q - \psi$ with $-\Delta\varphi = p_\alpha$, zero trace.



$k = 3$ on a pentagon:
• vertex • edge • interior.

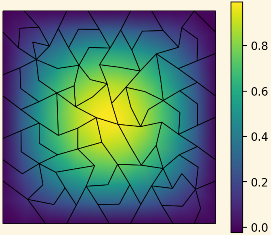
$\mathbb{P}_k \subset \hat{V}_h^k$ reproduced to machine precision.

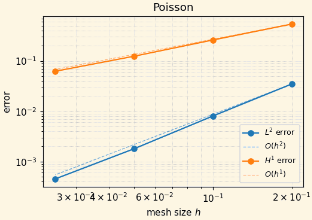
THE POISSON PROBLEM

$-\Delta u = f$ on $(0,1)^2$, $u = 0$ on $\partial\Omega$, for which the exact solution is given by $u = \sin \pi x \sin \pi y$.

```
fes = LightningVEMSpace(mesh,
order=1, polysize=4, dirichlet=".*")
u,v = fes.TnT()
a = BilinearForm(grad(u)*grad(v)*dx)
f = LinearForm(2*pi*pi*exact*v*dx)
gfu.vec.data = a.mat.Inverse(
fes.FreeDofs(),"sparsecholesky")*f.vec
```

Optimal $\|u - u_h\|_{L^2} = \mathcal{O}(h^2)$, $|u - u_h|_{H^1} = \mathcal{O}(h)$.

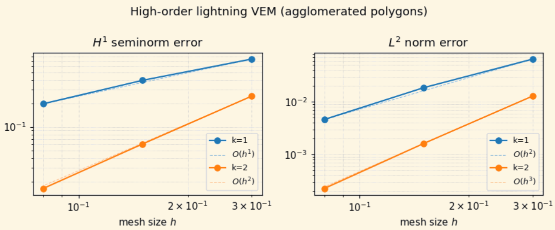




On agglomerated polygons the lightning VEM attains the optimal rates

$$|u-u_h|_{H^1}=\mathcal{O}(h^k),\quad \|u-u_h\|_{L^2}=\mathcal{O}(h^{k+1}).$$

Orders $k=1,2$ are optimal on agglomerated polygons. For $k=3$ it is optimal on clean elements but degrades on fine agglomerated meshes.

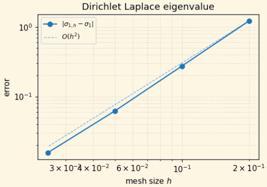


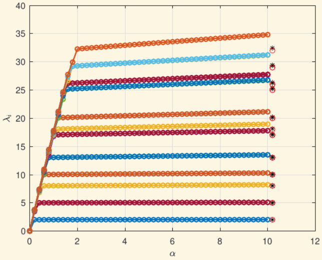
EIGENVALUES – NO SPURIOUS MODES

Consider the Dirichlet eigenproblem

$$-\Delta u = \lambda u \text{ on } \Omega, \; u = 0 \text{ along } \partial\Omega.$$

Classical VEM's **stabilization** pollutes the spectrum: as its strength α grows, eigenvalues drift off into **spurious** branches. The lightning VEM has *no* stabilization, so it stays on the exact values – with the same $\mathcal{O}(h^2)$ rate.





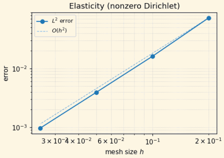
Eigenvalues vs. stabilization α (Trezzi & Zerbinati 2025). ○ exact, $*$ lightning VEM; the oblique branches are the stabilization-induced spurious modes.

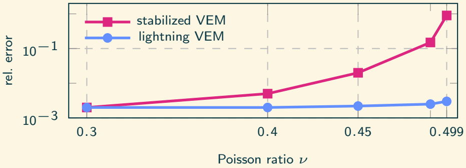
AN EXAMPLE FROM LINEAR ELASTICITY

We now consider the PDE associated with linear elasticity, i.e. $-\operatorname{div}\sigma(u) = f$, $\sigma = 2\mu\,\varepsilon(u) + \lambda\,\mathrm{tr}\,\varepsilon(u)\,I$, with *nonzero* Dirichlet data.

```
set_dirichlet(gfu, g)
```

The $k = 1$ degrees of freedom are exact corner values, so Dirichlet data is imposed by writing them directly, no `Set(BND)` needed. Optimal $\mathcal{O}(h^2)$ in L^2.





Illustrative: elasticity eigenvalue error as $\nu \to \frac{1}{2}$; lightning VEM stays **locking-free**.

Multigrid needs a prolongation P (coarse $\to$ fine). For classical VEM this is awkward: the degrees of freedom are **moments**, so a coarse basis value at a fine node is not directly available – and when you prolong you must even *reconstruct the interior moments* on the fine space.

The lightning VEM avoids this at $k = 1$: its degrees of freedom **are exact nodal point values**, and the per-polygon basis evaluates at *any* point. The prolongation is just

$$P[d_{\text{fine}}, d_{\text{coarse}}] = \varphi^{c}_{d_{\text{coarse}}}\left(x_{d_{\text{fine}}}\right).$$

One application is a symmetric multiplicative V-cycle:

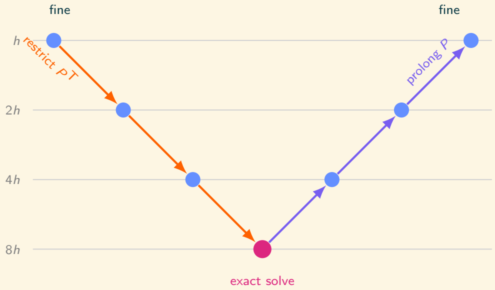
pre-smooth (symmetric Gauss–Seidel),

restrict the residual with P^{T},

recurse; exact sparse-Cholesky at the bottom,

prolong the correction with P, post-smooth.

SPD by construction $\Rightarrow$ a valid **CG preconditioner**; the figure of merit is the CG iteration count.



Smooth on the way down and up; solve exactly at the coarsest level.

POISSON MULTIGRID – MESH INDEPENDENCE

Geometric (VEM on every level):

levels	ndof	CG its	L^2 err
2	59	7	2.2e-02
3	210	8	5.8e-03
4	800	8	1.4e-03
5	3111	8	3.5e-04

Agglomeration (P1 fine, VEM coarse):

`maxh`	ndof_f	ndof_c	re-d.	Gal.	L^2 err
0.1	136	*57*	5	4	7.8e-03
0.05	511	*205*	6	5	1.8e-03
0.025	1945	*792*	5	5	4.3e-04
0.0125	7575	*3074*	5	5	1.1e-04

$\mathrm{ndof}_f/\mathrm{ndof}_c$: fine/coarse dofs; *coarse counts illustrative.*

Two coarse operators, differing in one object only:

re-discretize: assemble the VEM stiffness directly on the coarse polygons – a genuine standalone discretisation;

Galerkin: form it algebraically as $P^{\mathsf{T}}A\,P$ from the fine operator.

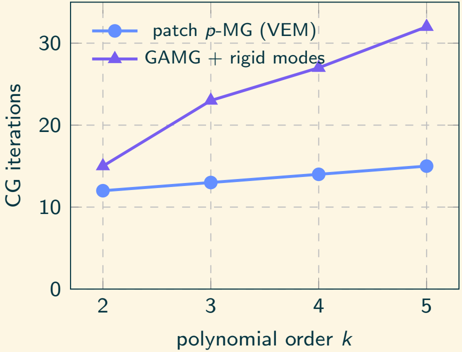
Agglomeration **reduces the degrees of freedom** on the coarse level, yet the CG count is **mesh independent** for both operators while L^2 accuracy is preserved.

ELASTICITY – VERTEX-PATCH p-MULTIGRID, VEM COARSE

High-order elasticity $V_h = [\mathbb{P}_k]^2$, preconditioned by a two-level V-cycle:

- **Fine:** damped **vertex-patch** smoother (additive Schwarz over the elements around each vertex);
- **Coarse:** p-coarsen to $\mathbb{P}_1$, then **lightning-VEM agglomeration**.

Galerkin per component: the scalar point-value prolongation is applied to each displacement component independently (block-diagonal P), and the coarse operators are $P^{\top} A P$ – so the elasticity coupling is carried down even though the VEM space is scalar.



Fixed mesh, raising k: the VEM patch p-MG is ***p*-robust**; GAMG grows.

WHAT NGSVEM GIVES YOU

The space

`LightningVEMSpace(mesh, order, polysize, dirichlet)`, a ready-to-use NGSolve FE space on agglomerated polygons. `polysize`, `beta`.

Multigrid

`MultigridPreconditioner`, point-value prolongations, `patch_vem_vcycle`, and PETSc helpers in `petsc_utils.py`.

```
pip install scikit-build-core
pip install --no-build-isolation .

# examples
python examples/poisson.py
python examples/eigenvalue.py
python examples/elasticity.py
python examples/poisson_multigrid.py
python examples/elasticity_patch_amg.py
```

Each example writes a webgui viewer, while theory, tutorials, API & multigrid are documented (Sphinx).

Oxford Mathematical Institute

1. A stabilization-free VEM

Rational "lightning" basis functions $\Rightarrow \int_K \nabla\varphi_i \cdot \nabla\varphi_j$ directly. No stabilization, no projection – implemented like an ordinary NGSolve element.

2. Pointwise access is the superpower

$k = 1$ degrees of freedom are exact nodal values $\Rightarrow$ a trivial, exact **point-value prolongation** that classical VEM cannot offer.

3. VEM as a coarse engine

The same agglomeration hierarchy preconditions Poisson (mesh independent) and high-order elasticity (p-robust), matching direct coarse solves.

REFERENCES

[1] M. L. Trezzi, U. Zerbinati. *When rational functions meet virtual elements: the lightning virtual element method.* Calcolo **61**, 35 (2024).
`https://link.springer.com/article/10.1007/s10092-024-00585-1`

[2] M. L. Trezzi, U. Zerbinati. *The High-Order Lightning Virtual Element Method.* In: Numerical Mathematics and Advanced Applications – ENUMATH 2023. Springer (2025).
`https://link.springer.com/chapter/10.1007/978-3-031-76591-9_22`

[3] M. L. Trezzi, U. Zerbinati. *The Lightning Virtual Element Method for Self-adjoint Eigenvalue Problems.* In: Numerical Mathematics and Advanced Applications – ENUMATH 2023. Springer (2025).
`https://link.springer.com/chapter/10.1007/978-3-031-76591-9_23`

[4] A. Gopal, L. N. Trefethen. *Solving Laplace problems with corner singularities via rational functions.* SIAM J. Numer. Anal. **57**(5), 2074–2094 (2019).
`https://people.maths.ox.ac.uk/trefethen/laplaceSINUM.pdf`

[5] ngsVEM documentation (theory, tutorials, API & multigrid).
`https://ngsvem.readthedocs.io/en/latest/`

THANK YOU!

ngsVEM: a lightning Virtual Element Method for NGSolve

Umberto Zerbinati[1]