# New developments in the CLAPP framework

E. Franck[1], M.Gaja[2], H. Guillard [7], M. Hölzl[2], A. Iaagoubi [7],K. Kormann[2],
J.Lakhlili[2], C. Manni[4], M.Mazza[2], B. Nkonga[3], A. Ratnani [2],
S. Serra-Capizzano[5], E. Sonnendrücker[2], H. Speleers[4], D. Toshniwal[6]

November 16, 2016

[1] Inria Nancy Grand Est and IRMA Strasbourg, France
[2] Max-Planck-Institut für Plasmaphysik, Garching, Germany
[3] University of Nice, France
[4] University of Rome Tor Vergata, Rome, Italy
[5] University of Insubria, Como, Italy
[6] ICES University of Texas, Austin, USA
[7] Inria Sophia-Antipolis, France

# Outline

- Motivations
- Preconditioning and GLT
- GLT for Harmonic Maxwell problem
- CLAPP: a framework for Computational Plasma Physics

# Act I

- Motivations
- Preconditioning and GLT
- GLT for Harmonic Maxwell problem

# Motivations

- Direct solvers are great but
  - have a complexity of $\mathcal{O}\left(n^{(d+1)/2}\right)$ using the sparsity of the matrix
  - memory limitation: the factorization (which is dense) cannot be stored for problems of interest
- Iterative solvers are good but
  - one has to deal with ill-conditioned matrices
  - ➠ needs preconditioners: algebraic, physics-based, etc
  - ➠ another alternative is to use the GLT, an elegant way of building preconditioners to fix a specific pathology

# Preconditioning: Problem setting

**Linear PDE:** $Au = b$

$\Downarrow$ **linear discretization method**

Sequence of linear systems $\{A_n u_n = b_n\}$ of increasing dimension $d_n$

The matrix $A_n$ may have a **structure**

**Example in 1d using Finite Differences:**

$$\begin{cases} -u'' = f & \text{in} \quad (0,1) \\ u = 0 & \text{on} \quad \partial(0,1) \end{cases} \quad \Rightarrow \quad A_n = \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -1 & 2 & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix}$$

i.e., $A_n$ is a so called Toeplitz matrix (constant along the diagonals)

# Preconditioning: Problem setting

- **Why structure is important?** Iterative methods, especially **multigrid** and **preconditioned Krylov** can exploit it in order to accelerate their convergence.

Their convergence depends on the **spectral features** of $A_n$

For structured matrices the spectral analysis is strictly related to the notion of **symbol**

**Qualitative definition:** the **symbol** is a function which describes the asymptotical spectral distribution of a matrix-sequence $\{A_n\}_n$

GLT sequences = a tool for computing spectral symbols

# Spectral tools: symbol

- **A little bit more accurate definition:**
  - $\{A_n\}_n$ = matrix-sequence, $\dim(A_n) = d_n \to \infty$
  - $f : D \subset \mathbb{R}^d \to \mathbb{C}, \quad 0 < \text{measure}(D) < \infty$

  $\{A_n\}_n$ has a **spectral distribution** described by $f$ means:

  The eigenvalues of $A_n$ are approximately a uniform sampling of $f$ over $D$.

  $f$ = **spectral symbol** of $\{A_n\}_n$. Notation: $\{A_n\}_n \sim_\lambda f$

- **E.g.:** When $d_n = n$, $d = 1$, $D = [0, \pi]$, $\{A_n\}_n \sim_\lambda f$ means

$$\lambda_j(A_n) \approx f\left(\frac{j\pi}{n}\right), \quad j = 0, \dots, n-1.$$

- **Remark:** this definition can also be given is the singular values sense (replacing $f \to |f|$). Notation: $\{A_n\}_n \sim_\sigma f$.

# Spectral tools: GLT theory

**The set of GLT sequences form a $*$-algebra (involutive algebra)**
i.e., it is closed under linear combinations, products, inversion, conjugation.

Let $\{A_\mathbf{n}\}_\mathbf{n} \sim_{GLT} \kappa_1$ and $\{B_\mathbf{n}\}_\mathbf{n} \sim_{GLT} \kappa_2$, then

- $\{\alpha A_\mathbf{n} + \beta B_\mathbf{n}\}_\mathbf{n} \sim_{GLT} \alpha\kappa_1 + \beta\kappa_2, \quad \alpha, \beta \in \mathbb{C}$;
- $\{A_\mathbf{n} B_\mathbf{n}\}_\mathbf{n} \sim_{GLT} \kappa_1\kappa_2$;
- if $\kappa_1$ vanishes, at most, in a set of zero Lebesgue measure, then $\{A_\mathbf{n}^{-1}\}_\mathbf{n} \sim_{GLT} \kappa_1^{-1}$;
- $\{A_\mathbf{n}^*\}_\mathbf{n} \sim_{GLT} \bar{\kappa}_1$.

➠ This $*$-algebra is not empty!

- $D_n(a)$, $a : [0,1] \to \mathbb{C}$ Riemann integrable function, a diagonal sampling matrix, i.e.,

$$D_n(a) = \begin{bmatrix} a(\frac{1}{n}) & & & \\ & a(\frac{2}{n}) & & \\ & & \ddots & \\ & & & a(1) \end{bmatrix}, \qquad \{D_n(a)\} \sim_\lambda a$$

# Spectral tools: GLT theory

**The set of GLT sequences form a $*$-algebra (involutive algebra)**
i.e., it is closed under linear combinations, products, inversion, conjugation.

Let $\{A_{\mathbf{n}}\}_{\mathbf{n}} \sim_{GLT} \kappa_1$ and $\{B_{\mathbf{n}}\}_{\mathbf{n}} \sim_{GLT} \kappa_2$, then

- $\{\alpha A_{\mathbf{n}} + \beta B_{\mathbf{n}}\}_{\mathbf{n}} \sim_{GLT} \alpha\kappa_1 + \beta\kappa_2, \quad \alpha, \beta \in \mathbb{C}$;
- $\{A_{\mathbf{n}} B_{\mathbf{n}}\}_{\mathbf{n}} \sim_{GLT} \kappa_1\kappa_2$;
- if $\kappa_1$ vanishes, at most, in a set of zero Lebesgue measure, then $\{A_{\mathbf{n}}^{-1}\}_{\mathbf{n}} \sim_{GLT} \kappa_1^{-1}$;
- $\{A_{\mathbf{n}}^*\}_{\mathbf{n}} \sim_{GLT} \bar{\kappa}_1$.

➠ This $*$-algebra is not empty!

- $T_n(f)$, i.e., a Toeplitz matrix obtained from the Fourier coefficients of $f : [-\pi, \pi] \to \mathbb{C}$, with $f \in L^1([-\pi, \pi])$ as follows

$$T_n(f) = \begin{bmatrix} f_0 & f_{-1} & \cdots & f_{-(n-1)} \\ f_1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & f_{-1} \\ f_{n-1} & \cdots & f_1 & f_0 \end{bmatrix}, \qquad \{T_n(f)\} \sim_\lambda f$$

Let's summarize,

- we can construct a $*$-algebra to *mimic* the eigenvalues of sequence of matrices.
- But this is not sufficient to *capture* the spectral behavior of a B-Splines discretization!

**Solution**

- Enrich the $*$-algebra with terms like $\int_\Omega \mathcal{D}^{(r)}\varphi_i \mathcal{D}^{(s)}\varphi_j$. But, how?
- Simply, buy computing their exact symbol (or an approximation c.f. later for Maxwell)

**Example: Mass matrix** $\int_0^1 N_i^p N_j^p$

$$m_p(x,\theta) := m_p(\theta) = \phi_{2p+1}(p+1) + 2\sum_{k=1}^{p} \phi_{2p+1}(p+1-k)\cos(k\theta). \tag{1}$$

**Example: Stiffness matrix** $\int_0^1 \left(N_i^p\right)' \left(N_j^p\right)'$

$$s_p(x,\theta) := s_p(\theta) = -\phi''_{2p+1}(p+1) - 2\sum_{k=1}^{p} \phi''_{2p+1}(p+1-k)\cos(k\theta). \tag{2}$$

where $\phi_{2p+1}$ is the cardinal B-Spline of degree $2p+1$

- In 2d and 3d, we can use the previous symbols and Kronecker algebra
- Are we limited to linear problems? ➠ No!

**Example** Let's consider the following weak formulation

$$D_{ij}(\alpha, \boldsymbol{\beta}, \epsilon) = \left( \int_\Omega \alpha \varphi_j \varphi_i + \beta_1 \varphi_j \partial_x \varphi_i + \beta_2 \varphi_j \partial_y \varphi_i + (\partial_x \beta_1 + \partial_y \beta_2) \varphi_j \varphi_i + \epsilon \nabla \varphi_i \cdot \nabla \varphi_j \right)$$

The symbol of the associated sequence of linear system is

$$\begin{aligned} d_p(\alpha, \boldsymbol{\beta}, \epsilon, h; \mathbf{x}, \boldsymbol{\theta}) := & \ \alpha m_p(\theta_1) m_p(\theta_2) \\ & + h \left(\beta_1(\mathbf{x}) a_p(\theta_1) m_p(\theta_2) + \beta_2(\mathbf{x}) m_p(\theta_1) a_p(\theta_2)\right) \\ & + h \left(\partial_x \beta_1(\mathbf{x}) + \partial_y \beta_2(\mathbf{x})\right) m_p(\theta_1) m_p(\theta_2) \\ & + \epsilon h^2 \left(s_p(\theta_1) m_p(\theta_2) + m_p(\theta_1) s_p(\theta_2)\right) \end{aligned}$$

# Spectral tools: GLT theory

## Fundamental property

Each GLT sequence $\{A_n\}_n$ is equipped with a symbol in the singular value sense, i.e. there exists a function $\chi : [0,1] \times [-\pi, \pi] \to \mathbb{C}$ such that

$$\{A_n\}_n \sim_\sigma \chi$$

**E.g.:** if $A_n = D_n(a) T_n(f)$, then $\{A_n\}_n \sim_\sigma \chi = a \cdot f$

**Advantage of this tool**: studying the symbol

- we retrieve information on the conditioning
- we get hints on how to design good preconditioning strategies, because of this property: if $\{A_n\}_n \sim_\sigma f$ and $\{B_n\}_n \sim_\sigma g$, then

$$\{B_n^{-1} A_n\}_n \sim_\sigma g^{-1} f$$

Target: choose $g$ in order to eliminate the 'pathologies' of $f$

➠ c.f. M. Gaja talk for Poisson

- $s_p$ is nonnegative and has a unique zero in 0 of order 2 $\Rightarrow$ $n^{d-2} L_n$ is ill-conditioned in the low frequencies. Classical problem solved by MG preconditioning.
- $s_p$ has infinitely many exponential zeros at the $\pi$-edges when $p$ becomes large $\Rightarrow$ $n^{d-2} L_n$ is ill-conditioned in the high frequencies. Non-canonical problem solvable by GLT theory.

# GLT for curl-curl problem

- **Application**: compatible B-Splines discretization based on the discrete De Rham sequence of this variational problem:

Find $\mathbf{u} \in H(\text{curl}, \Omega)$ such that

$$(\vec{\nabla} \times \mathbf{u}, \vec{\nabla} \times \mathbf{v}) + \nu\,(\mathbf{u}, \mathbf{v}) = (\mathbf{f}, \mathbf{v}), \quad \forall \mathbf{v} \in H(\text{curl}, \Omega),$$

where $\nu \geq 0$ and $H(\text{curl}, \Omega) := \{\mathbf{u} \in (L^2([0,1]^2))^2 \text{ s.t. } \vec{\nabla} \times \mathbf{u} \in L^2([0,1]^2)\}$.

- **Coefficient matrix** $\mathcal{A}_n^{\nu}$: is a $2 \times 2$ block matrix.

- **Spectral symbol** $f^{\nu}$:
  - 2D problem $\Rightarrow$ $f^{\nu}$ is bivariate (defined in $[-\pi, \pi]^2$);
  - vectorial problem $\Rightarrow$ $f^{\nu}$ is $2 \times 2$ matrix-valued function. In this case, we have to look at the two eigenvalue functions of $f^{\nu}$.

# GLT for curl-curl problem

## Eigenvalue functions of $f^\nu$

$$\lambda_1\left(f^\nu(\theta_1,\theta_2)\right) \approx m_{p-1}(\theta_1) m_{p-1}(\theta_2) \frac{\nu}{n^2}$$

$$\lambda_2\left(f^\nu(\theta_1,\theta_2)\right) \approx m_{p-1}(\theta_1) m_{p-1}(\theta_2) \left[4 - 2\cos(\theta_1) - 2\cos(\theta_2) + \frac{\nu}{n^2}\right]$$

**A nice connection between continuous problem and spectral information**:

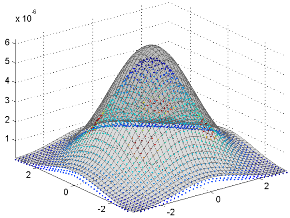
- **Continuum**: the curl-curl operator has infinite dimensional kernel and on the complement behaves as a second order operator.

$$\Downarrow$$

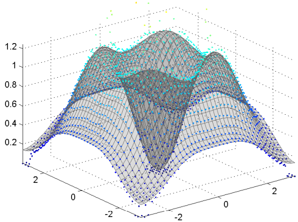
- **Spectral counterpart**: when $\nu = 0$, $\lambda_1(f^\nu) \equiv 0$, while $\lambda_2(f^\nu)$ is the symbol of the 2D Laplacian operator.

- **Ok, nice...but what can we do with this information?**

An equispaced sampling of the eigenvalues functions in $[-\pi,\pi]^2$ gives an approximation of the eigenvalues of $\mathcal{A}_{\mathbf{n}}^{\nu}$.




$\lambda_1(f^{\nu})$




$\lambda_2(f^{\nu})$

Comparison between the eigenvalues of $\mathcal{A}_{\mathbf{n}}^{\nu}$ (colored dots) and $\lambda_k(f^{\nu})$, $k=1,2$, when $n=40$, $p=3$, $\nu=10^{-2}$ (matrix-size 3612).

# GLT for curl-curl problem

**A study of the eigenvalue functions tell us that:**

(1) $\mathcal{A}_n^\nu$ is ill-conditioned in the low frequencies. Classical problem solved by MG preconditioning.

(2) $\mathcal{A}_n^\nu$ is ill-conditioned in the high frequencies. Non-canonical problem solvable by GLT theory.

- **Solver proposal**: Using the symbol we can construct a smoother for MG valid for high-frequencies:

PCG or the PGMRES with preconditioner

$$I_2 \otimes T(m_{p-1}(\theta_1)) \otimes T(m_{p-1}(\theta_2))$$

- **Remark:** such a preconditioner is a tensor product of banded matrices then only a linear computational cost is required.

# Construction of the Multigrid

- Use the Auxiliary Space Preconditioning method[1]
- Proposed preconditioner (HX): $R + \mathcal{G} B_h \mathcal{G}^T + \mathbf{\Pi}_h^{\text{curl}} \mathbf{B}_h \left( \mathbf{\Pi}_h^{\text{curl}} \right)^T$
  where
  - $B_h$ correponds to MultiGrid V-cycles solver for the poisson problem $(\nabla u, \nabla v) + \mu(u, v)$
  - $\mathbf{B}_h$ correponds to MultiGrid V-cycles solver for the poisson problem $(\nabla \mathbf{u}, \nabla \mathbf{v}) + \mu(\mathbf{u}, \mathbf{v})$
- How to construct the operators $\mathbf{\Pi}_h^{\text{grad}}$ and $\mathbf{\Pi}_h^{\text{curl}}$?
  - use the projection-based interpolation by Demkovicz? (in progress)

.

[1] Hiptmair, Xu, *SIAM J. Numer. Anal.*, 2007

# DeRham sequence

## The continuous case

here without boundary conditions

$$\mathbb{R} \hookrightarrow H^1(\Omega) \xrightarrow{\quad \boldsymbol{\nabla} \quad} H(\text{curl}, \Omega) \xrightarrow{\quad \vec{\nabla}\times \quad} H(\text{div}, \Omega) \xrightarrow{\quad \nabla\cdot \quad} L^2(\Omega) \longrightarrow 0 \qquad (3)$$

using **pullbacks** in the case of a mapping (vector fields transformations)

$$\begin{array}{ccccccc} H^1(\Omega) & \xrightarrow{\quad \boldsymbol{\nabla} \quad} & H(\text{curl}, \Omega) & \xrightarrow{\quad \vec{\nabla}\times \quad} & H(\text{div}, \Omega) & \xrightarrow{\quad \nabla\cdot \quad} & L^2(\Omega) \\ \imath^0 \uparrow & & \imath^1 \uparrow & & \imath^2 \uparrow & & \imath^3 \uparrow \\ H^1(\mathcal{P}) & \xrightarrow{\quad \boldsymbol{\nabla} \quad} & H(\text{curl}, \mathcal{P}) & \xrightarrow{\quad \vec{\nabla}\times \quad} & H(\text{div}, \mathcal{P}) & \xrightarrow{\quad \nabla\cdot \quad} & L^2(\mathcal{P}) \end{array} \qquad (4)$$

Commutative diagram between continuous and discrete spaces.

$$\begin{array}{ccccccc} H^1(\Omega) & \xrightarrow{\quad \boldsymbol{\nabla} \quad} & H(\text{curl}, \Omega) & \xrightarrow{\quad \vec{\nabla}\times \quad} & H(\text{div}, \Omega) & \xrightarrow{\quad \nabla\cdot \quad} & L^2(\Omega) \\ \boldsymbol{\Pi}_h^{\text{grad}} \downarrow & & \boldsymbol{\Pi}_h^{\text{curl}} \downarrow & & \boldsymbol{\Pi}_h^{\text{div}} \downarrow & & \boldsymbol{\Pi}_h^{L^2} \downarrow \\ V_h(\text{grad}, \Omega) & \xrightarrow{\quad \boldsymbol{\nabla} \quad} & V_h(\text{curl}, \Omega) & \xrightarrow{\quad \vec{\nabla}\times \quad} & V_h(\text{div}, \Omega) & \xrightarrow{\quad \nabla\cdot \quad} & V_h(L^2, \Omega) \end{array} \qquad (5)$$

Buffa et al[2009] show the construction of a discrete DeRham sequence using B-Splines.

$$\mathbb{R} \hookrightarrow \underbrace{\mathcal{S}^{p,p,p}}_{V_h(\text{grad},\mathcal{P})} \xrightarrow{\quad \nabla \quad} \underbrace{\begin{pmatrix} \mathcal{S}^{p-1,p,p} \\ \mathcal{S}^{p,p-1,p} \\ \mathcal{S}^{p,p,p-1} \end{pmatrix}}_{V_h(\text{curl},\mathcal{P})} \xrightarrow{\quad \vec{\nabla}\times \quad} \underbrace{\begin{pmatrix} \mathcal{S}^{p,p-1,p-1} \\ \mathcal{S}^{p-1,p,p-1} \\ \mathcal{S}^{p-1,p-1,p} \end{pmatrix}}_{V_h(\text{div},\mathcal{P})} \xrightarrow{\quad \nabla\cdot \quad} \underbrace{\mathcal{S}^{p-1,p-1,p-1}}_{V_h(L^2,\mathcal{P})} \longrightarrow 0 \tag{6}$$

$$\begin{array}{ccccccc}
\mathcal{C}^\infty(\Omega) & \xrightarrow{\quad \nabla \quad} & \mathcal{C}^\infty(\Omega) & \xrightarrow{\quad \vec{\nabla}\times \quad} & \mathcal{C}^\infty(\Omega) & \xrightarrow{\quad \nabla\cdot \quad} & \mathcal{C}^\infty(\Omega) \\
\Pi_h^{\text{grad}} \big\downarrow & & \Pi_h^{\text{curl}} \big\downarrow & & \Pi_h^{\text{div}} \big\downarrow & & \Pi_h^{L^2} \big\downarrow \\
V_h(\text{grad},\Omega) & \xrightarrow{\quad \nabla \quad} & V_h(\text{curl},\Omega) & \xrightarrow{\quad \vec{\nabla}\times \quad} & V_h(\text{div},\Omega) & \xrightarrow{\quad \nabla\cdot \quad} & V_h(L^2,\Omega)
\end{array} \tag{7}$$

# DeRham sequence

## Discrete case for B-Splines: The $1D$ case

- DeRham sequence is reduced to $$\mathbb{R} \hookrightarrow \underbrace{\mathcal{S}^p}_{V_h(\text{grad},\mathcal{P})} \xrightarrow{\nabla} \underbrace{\mathcal{S}^{p-1}}_{V_h(L^2,\mathcal{P})} \longrightarrow 0$$
- The recursion formula for derivative writes

$$N_i^{p\prime}(t) = D_i^p(t) - D_{i+1}^p(t) \quad \text{where} \quad D_i^p(t) = \frac{p}{t_{i+p+1} - t_i} N_i^{p-1}(t)$$

- we have $\mathcal{S}^{p-1} = \mathbf{span}\{N_i^{p-1}, 1 \leq i \leq n-1\} = \mathbf{span}\{D_i^p, 1 \leq i \leq n-1\}$
  ➠ a change of basis as a diagonal matrix
- Now if $u \in S^p$, with and expansion $u = \sum_i u_i N_i^p$, we have

$$u' = \sum_i u_i \left(N_i^p\right)' = \sum_i (-u_{i-1} + u_i) D_i^p$$

- If we introduce the B-Splines coefficients vector $\mathbf{u} := (u_i)_{1 \leq i \leq n}$ (and $\mathbf{u}^\star$ for the derivative), we have

$$\mathbf{u}^\star = D\mathbf{u}$$

where $D$ is the incidence matrix (of entries $-1$ and $+1$)

$$
\begin{array}{ccccccc}
H^1(\Omega) & \xrightarrow{\nabla} & H(\text{curl},\Omega) & \xrightarrow{\vec{\nabla}\times} & H(\text{div},\Omega) & \xrightarrow{\nabla\cdot} & L^2(\Omega) \\
\Pi_h^{\text{grad}} \downarrow & & \Pi_h^{\text{curl}} \downarrow & & \Pi_h^{\text{div}} \downarrow & & \Pi_h^{L^2} \downarrow \\
V_h(\text{grad},\Omega) & \underset{\mathcal{G}^T}{\overset{\mathcal{G}}{\rightleftharpoons}} & V_h(\text{curl},\Omega) & \underset{\mathcal{C}^T}{\overset{\mathcal{C}}{\rightleftharpoons}} & V_h(\text{div},\Omega) & \underset{\mathcal{D}^T}{\overset{\mathcal{D}}{\rightleftharpoons}} & V_h(L^2,\Omega)
\end{array}
\tag{8}
$$

Let $I$ be the identity matrix, we have in the $2D$ case:

$$
\mathcal{G} = \begin{pmatrix} D \otimes I \\ I \otimes D \end{pmatrix} \tag{9}
$$

$$
\mathcal{C} = \begin{pmatrix} I \otimes D \\ -D \otimes I \end{pmatrix} \quad \text{[scalar curl]}, \quad \mathcal{C} = \begin{pmatrix} -I \otimes D & D \otimes I \end{pmatrix} \quad \text{[vectorial curl]} \tag{10}
$$

$$
\mathcal{D} = \begin{pmatrix} D \otimes I & I \otimes D \end{pmatrix} \tag{11}
$$

## Discrete derivatives for B-Splines

$$
\begin{array}{ccccccc}
H^1(\Omega) & \xrightarrow{\nabla} & H(\text{curl}, \Omega) & \xrightarrow{\vec{\nabla}\times} & H(\text{div}, \Omega) & \xrightarrow{\nabla\cdot} & L^2(\Omega) \\
\Pi_h^{\text{grad}} \downarrow & & \Pi_h^{\text{curl}} \downarrow & & \Pi_h^{\text{div}} \downarrow & & \Pi_h^{L^2} \downarrow \\
V_h(\text{grad}, \Omega) & \underset{\mathcal{G}^T}{\overset{\mathcal{G}}{\rightleftharpoons}} & V_h(\text{curl}, \Omega) & \underset{\mathcal{C}^T}{\overset{\mathcal{C}}{\rightleftharpoons}} & V_h(\text{div}, \Omega) & \underset{\mathcal{D}^T}{\overset{\mathcal{D}}{\rightleftharpoons}} & V_h(L^2, \Omega)
\end{array} \tag{8}
$$

Let $I$ be the identity matrix, we have in the $3D$ case:

$$
\mathcal{G} = \begin{pmatrix} D \otimes I \otimes I \\ I \otimes D \otimes I \\ I \otimes I \otimes D \end{pmatrix} \tag{12}
$$

$$
\mathcal{C} = \begin{pmatrix} 0 & -I \otimes I \otimes D & I \otimes D \otimes I \\ I \otimes I \otimes D & 0 & -D \otimes I \otimes I \\ -I \otimes D \otimes I & D \otimes I \otimes I & 0 \end{pmatrix} \tag{13}
$$

$$
\mathcal{D} = \begin{pmatrix} D \otimes I \otimes I & I \otimes D \otimes I & I \otimes I \otimes D \end{pmatrix} \tag{14}
$$

# Conclusion and perspectives

**Summary**

- We use the GLT theory to spectrally analyse matrices coming from a IgA discretization of the curl-curl problem.
- We exploit the obtained spectral information to suggest a suitable solver for the corresponding linear systems.

**Ongoing work and Perspectives**

- projectors based interpolation to ensure the commutativity of the discrete DeRham sequence.
- 3D case
- Application for Tokamak Plasma

$$\vec{\nabla} \times \vec{\nabla} \times \mathbf{E} - \frac{\omega^2}{c^2} K \, \mathbf{E} = \mathbf{f}$$

where

$$K = \begin{pmatrix} S & -iD & 0 \\ iD & S & 0 \\ 0 & 0 & P \end{pmatrix} + \frac{i}{\epsilon_0 \omega} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & g \end{pmatrix}$$

# Act II: CLAPP–a framework for Computational Plasma Physics

- Efficient 6d Vlasov–Poisson solver
- Geometric electromagnetic PIC framework
- Finite Elements in CLAPP: Jorek-Django

# CLAPP Framework: Motivations

- As a user, you want a fast code and you want it now
- As a developer, you want to code fast code, faster

# CLAPP Framework: Motivations

- As a user, you want a fast code and you want it now
- As a developer, you want to code fast code, faster

# CLAPP Framework: Motivations

- As a user, you want a fast code and you want it now
- As a developer, you want to code fast code, faster

- Within CLAPP, we try to offer robust numerical methods allowing researchers to build complicated simulations.

- As a user, you want a fast code and you want it now
- As a developer, you want to code fast code, faster


- Within CLAPP, we try to offer robust numerical methods allowing researchers to build complicated simulations.

➠ It's not easy!

# CLAPP Framework: Available libraries

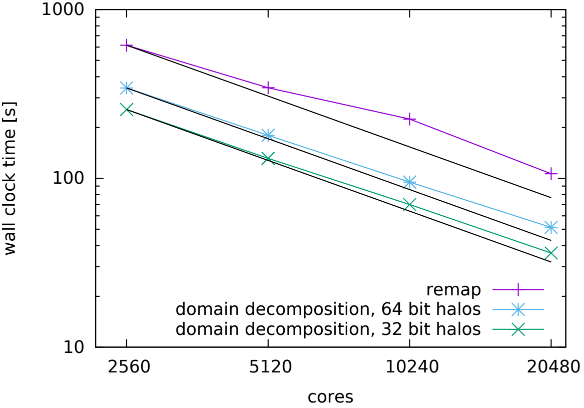
- **CLAPPIO** Input/Output Library
- **PLAF** Parallel Linear Algebra Library
- **SPL** Library for NURBS/B-Splines
- **DISCO** Abstract Discretization Context Library
- **FEMA** Library of Finite Elements Assemblers
- **HYPI** A PIC for Hybrid pushers based on pp forms
- **GLT** Library of Preconditioners and linear solver for B-Splines discretizations
- **SPIGA** Structure Preserving IsoGeoemtric Analysis library. Implements specific models (poisson, . . . )
- **SELALIB** Library of Semi-Lagrangian (and PIC methods)
- **CIMEQ** Common interface for magnetic equilibria

# SELALIB: Efficient 6d Vlasov–Poisson solver

- **Numerics**: Semi-Lagrangian solver with Lagrange or spline interpolation.
- **Parallelization schemes**
  - Domain partitioning into 6d cubes with adapted interpolation schemes.
  - Remap between two domain partitionings: One keeping **x** sequential and one keeping **v** sequential.
- **Optimizations**:
  - Vectorization of interpolation routines.
  - Cache-efficient memory layout.
- **Computing**
  - Strong scaling of about 90% efficiency from 2560 to 20480 cores.
  - Ported to new Intel Knights Landing architecture with performance comparable to Intel Xeon E5-2698 node.

# SELALIB: Efficient 6d Vlasov–Strong scaling

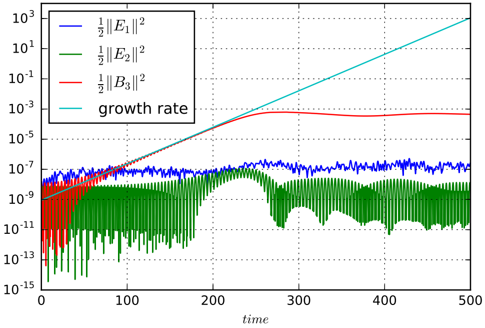
**Configuration**: $64^6$ grid points, 50 time steps, 7-point Lagrange interpolation, 4 MPI with 5OMP-threads per node.
**Hardware**: Ivy Bridge (hydra@mpcdf) (64 GB per node, InfiniBand FDR14).



# SELALIB: Geometric electromagnetic PIC framework

- **Discretization**: Conforming spline finite elements for fields (discrete deRham complex), Particle–In–Cell for distribution functions.
- **Formulation** of equations based on semi-discrete Hamiltonian and Poisson bracket.
- **Temporal discretizations**:
  - Symplectic method based on Hamiltonian splitting.
  - Average vector field splitting method: Semi-implicit (only implicit in field equations), energy conserving.

# SELALIB: Weibel instability 1d2v: Conservation properties.




| Propagator | total energy | Gauss law | momentum $P_2$ |
|---|---|---|---|
| Hamiltonian | 6.3E-7 | 1.5E-14 | 3.2E-15 |
| Boris | 3.4E-10 | 1.0E-4 | 1.3E-14 |
| AVF | 1.2E-16 | 3.8E-7 | 2.1E-14 |

# Finite Elements in CLAPP: Jorek-Django

- a collection of libraries written in Fortran2003
- these libraries are part of a more general framework (CLAPP) for computational plasma physics, developped at the NMPP.
- **Important features**
  - Parallel using MPI (+ OpenMP in progress)
  - compatible Finite elements discretizations for $H^1(\Omega), H(\text{curl}, \Omega), H(\text{div}, \Omega), L^2(\Omega)$
  - Collocation method in $1D$, *i.e.* toroidal direction, (in progress)
  - Isoparametric/Isogeometric + Standard discretizations
  - General boundary conditions (including strong/weak ones)
  - Matrix-Free for nonlinear problems
  - Physics-Based preconditioning
  - Auxiliary Spaces Preconditioning (in progress)
  - Multilevel methods, for B-Splines
  - Robust Multigrid for B-Splines based on the GLT theory (in progress)
    - Poisson and $H^1$-elliptic problems
    - $H(\text{curl})$ and $H(\text{div})$-elliptic problems

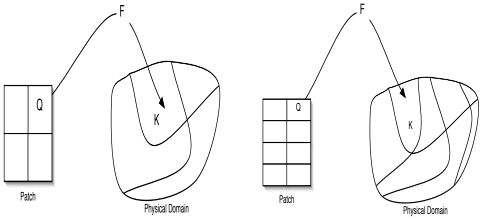
# Applications

Some examples solved using Jorek-Django

- Geometric Multigrid for B-Splines
  - Poisson (Implemented)
  - Maxwell (in progress)
- Helmoltz equation
- MHD equilibrium
- Anisotropic Diffusion
- Harmonic Domain Maxwell and Full-wave (in progress)
- Time Domain Maxwell (in progress)
- Reduced MHD (under validation)
- Physics-Based preconditioning for the wave equation
- Physics-Based preconditioning for the $3D$ reduced MHD (under validation)
- Burger and Euler using a relaxation method (validated in 1d)

# Discretization

## The IsoGeometric Approach




**Grid generation:** the use of $h/p/k$-refinement keeps the mapping **F** unchanged.

- Compact support
- Partition of Unity
- Affine covariance
- IsoParametric concept
- Error estimates in Sobolev norms
- Exacte DeRham discrete sequence

# Jorek-Django: Conclusion and perspectives

**Conclusions**

- we have developped a Parallel framework for Finite Elements for $H^1(\Omega), H(\text{curl}, \Omega), H(\text{div}, \Omega)$ problems
- B-Splines discretizations are fully validated
- First (internal) Pre-Release expected before February 2017

**Ongoing work and Perspectives**

- new quadrature rules for B-Splines: reduces the number of points per element
  - ➠ well adapted to uniform unclamped B-Splines
  - ➠ needs Nitsche method to impose the boundary condition
- other discretizations still in progress
- Physics-Based Preconditioner for the Reduced-MHD (model199 then 303)
- OpenMP, OpenACC
- mesh generation
  - ➠ Alignement and equidistributed meshes
  - ➠ $\mathcal{C}^1$ constraints in polar-like meshes and X-point using a local construction for arbitrary regularity for tensor B-Splines.

**Statistics**

- number of commits: $2'215$
- number of lines: $76'225$ (not including models $\sim 30'000$)
- documentation: about 400 pages (and more to come)

JorekDjango is the association of a set of libraries from CLAPP that allows the user to write (system of) partial differential equations and solve them using a Finite Element or Collocation method.

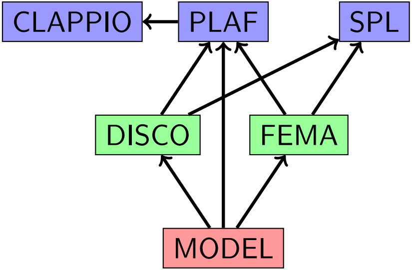



Figure : Strucutre of the JorekDjango Framework

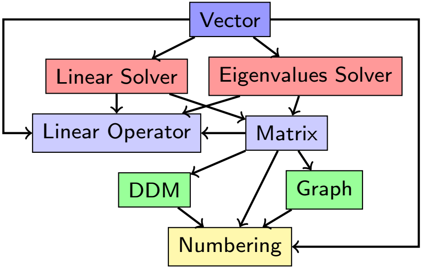
### Linear Algebra Objects

- Linear Operator
- Matrix
- Linear Solver
- Eigenvalues Solver
- Vector

### Discretization Objects

- Numbering
- Graph
- DDM

Internal PLAF dependencies



To create a family of *B-splines*, we need a non-decreasing sequence of knots $T = (t_i)_{1 \leqslant i \leqslant N+k}$, also called **knot vector**, with $k = p + 1$.
Each set of knots $T_j = \{t_j, \cdots, t_{j+p}\}$ will generate a *B-spline* $N_j$.

## Definition (B-Spline serie)

The j-th B-Spline of order $k$ is defined by the recurrence relation:

$$N_j^k = w_j^k N_j^{k-1} + (1 - w_{j+1}^k) N_{j+1}^{k-1}$$

where,

$$w_j^k(x) = \frac{x - t_j}{t_{j+k-1} - t_j} \qquad N_j^1(x) = \chi_{[t_j, t_{j+1}[}(x)$$

for $k \geq 1$ and $1 \leq j \leq N$.

# B-Splines Discretization: Knot vector families

## Clamped knots

**uniform**

$$T_1 = \{0, 0, 0, 1, 2, 3, 4, 5, 5, 5\}$$
$$T_2 = \{-0.2, -0.2, 0.0, 0.2, 0.4, 0.6, 0.8, 0.8\}$$

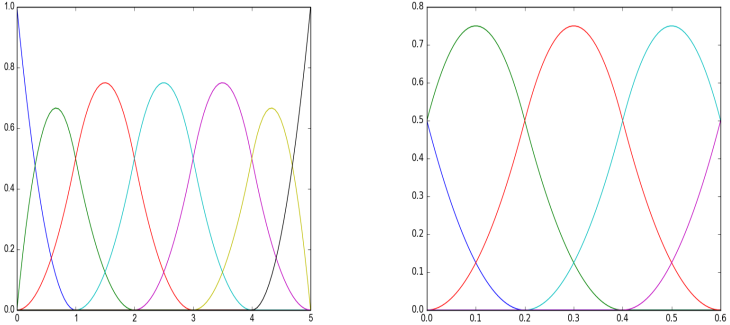



Figure : Quadratic B-Splines generated by $T_1$ (left) and $T_2$ (right)

Clamped knots

**non-uniform**

$$T_3 = \{0, 0, 0, 1, 3, 4, 5, 5, 5\}$$
$$T_4 = \{-0.2, -0.2, 0.4, 0.6, 0.8, 0.8\}$$

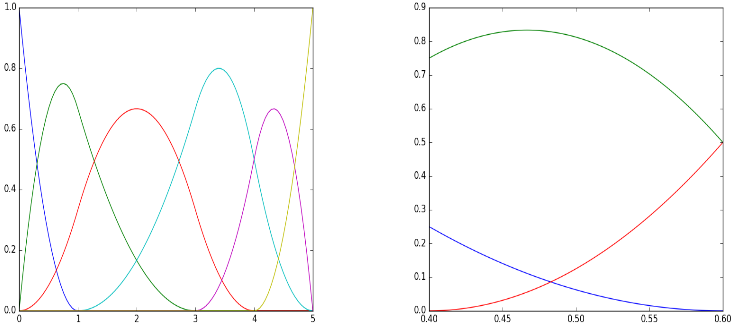



Figure : Quadratic B-Splines generated by $T_3$ (left) and $T_4$ (right)

## Unclamped knots

**uniform**

$$T_5 = \{0, 1, 2, 3, 4, 5, 6, 7\}$$
$$T_6 = \{-0.2, 0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$$

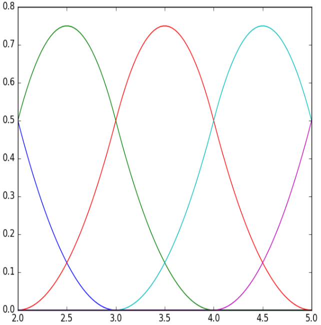



Figure : Quadratic B-Splines generated by $T_5$ (left) and $T_6$ (right)

## Unclamped knots

**non-uniform**

$$T_7 = \{0, 0, 3, 4, 7, 8, 9\}$$
$$T_8 = \{-0.2, 0.2, 0.4, 0.6, 1.0, 2.0, 2.5\}$$

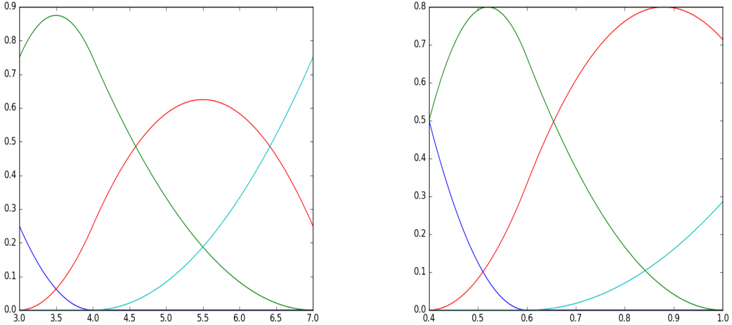



Figure : Quadratic B-Splines generated by $T_7$ (left) and $T_8$ (right)

## Refinement strategies

Refining the grid can be done in 3 different ways. This is the most interesting aspects of B-splines basis.

- **h-refinement** by inserting new knots. It is the equivalent of mesh refinement of the classical finite element method.
- **p-refinement** by elevating the B-spline degree. It is the equivalent of using higher finite element order in the classical FEM.
- **k-refinement** by increasing / decreasing the regularity of the basis functions (increasing / decreasing multiplicity of inserted knots).
- **r-refinement** moving the control points to reduce a given error estimate

# Reduce MHD model

## Single fluid resistive MHD

$$
\begin{cases}
\partial_t \rho + \nabla \cdot (\rho \mathbf{v}) = 0, \\
\rho \partial_t \mathbf{v} + \rho \mathbf{v} \cdot \nabla \mathbf{v} + \nabla p = \mathbf{J} \times \mathbf{B} - \nabla \cdot \overline{\overline{\mathbf{\Pi}}}, \\
\partial_t p + \mathbf{v} \cdot \nabla p + p \nabla \cdot \mathbf{v} + \nabla \cdot \mathbf{q} = 0 \\
\partial_t \mathbf{B} = -\nabla \times (-\mathbf{v} \times \mathbf{B} + \eta \mathbf{J}), \\
\nabla \cdot \mathbf{B} = 0, \quad \nabla \times \mathbf{B} = \mathbf{J}.
\end{cases}
$$

- **Reduced MHD model**: Reduce the number of variables and eliminate the fast waves in the reduced MHD model.
- We consider the cylindrical coordinate $(R, Z, \phi) \in \Omega \times [0, 2\pi]$.

## Reduced MHD: Assumption

$$
\mathbf{B} = \frac{F_0}{R}\mathbf{e}_\phi + \frac{1}{R}\nabla \psi \times \mathbf{e}_\phi, \quad \mathbf{v} = -R\nabla u \times \mathbf{e}_\phi + v_{||}\mathbf{B}
$$

with $u$ the electrical potential, $\psi$ the magnetic poloidal flux, $v_{||}$ the parallel velocity.

- **Initialization**: we use $\psi$ and pressure equilibrium, a zero velocity $(u = v_{||} = 0)$.
- **Wave structure**: low Mach and low $\beta$ regime $\rightarrow$ a large ratio between wave speeds.
- This problem coupled with hyperbolic structure generate ill-conditioned problem.

# Preconditioning

- The implicit system after linearization is given by

$$\begin{pmatrix} \mathbf{B}^{n+1} \\ p^{n+1} \\ \mathbf{u}^{\mathbf{n+1}} \end{pmatrix} = \begin{pmatrix} A_{\mathbf{B},p} & C_{\mathbf{B},p,\mathbf{u}} \\ C_{\mathbf{u},\mathbf{B},p} & A_{\mathbf{u}} \end{pmatrix}^{-1} \begin{pmatrix} R_{\mathbf{B}} \\ R_p \\ R_{\mathbf{u}} \end{pmatrix}$$

- with $A_{\mathbf{B},p}$ and $A_{\mathbf{u}}$ the advection terms linked to $\mathbf{B}$ and $p$ (resp $\mathbf{u}$), $C_{\mathbf{B},p,\mathbf{u}}$ and $C_{\mathbf{u},\mathbf{B},p}$ the coupling terms which gives the Alfven and acoustic waves.
- The solution of the system is given by

$$\begin{pmatrix} \mathbf{B}^{n+1} \\ p^{n+1} \\ \mathbf{u}^{\mathbf{n+1}} \end{pmatrix} = \begin{pmatrix} I_d & A_{\mathbf{B},p}^{-1} C_{\mathbf{B},p,\mathbf{u}} \\ 0 & I_d \end{pmatrix} \begin{pmatrix} A_{\mathbf{B},p}^{-1} & 0 \\ 0 & P_{schur}^{-1} \end{pmatrix} \begin{pmatrix} I_d & 0 \\ -C_{\mathbf{u},\mathbf{B},p} A_{\mathbf{B},p}^{-1} & I_d \end{pmatrix} \begin{pmatrix} R_{\mathbf{B}} \\ R_p \\ R_{\mathbf{u}} \end{pmatrix}$$

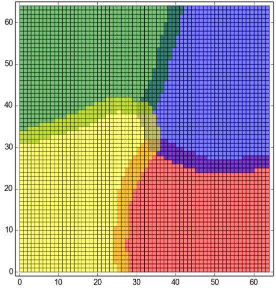
- Using the previous Schur decomposition, we obtain the following algorithm:

$$\begin{cases} \text{Predictor}: & A_{\mathbf{B},p} \begin{pmatrix} \mathbf{B}^* \\ p^* \end{pmatrix} = \begin{pmatrix} R_{\mathbf{B}} \\ R_p \end{pmatrix} \\ \text{Velocity evolution}: & P_{schur} \mathbf{u}^{n+1} = \left( -C_{\mathbf{u},\mathbf{B},p} \begin{pmatrix} \mathbf{B}^{n+1} \\ p^{n+1} \end{pmatrix} + R_{\mathbf{u}} \right) \\ \text{Corrector}: & A_{\mathbf{B},p} \begin{pmatrix} \mathbf{B}^{n+1} \\ p^{n+1} \end{pmatrix} = A_{\mathbf{B},p} \begin{pmatrix} \mathbf{B}^* \\ p^* \end{pmatrix} - C_{\mathbf{B},p,\mathbf{u}} \mathbf{u}_{n+1} \end{cases}$$

- **Preconditioning**: we approximate the Schur complement by a multi-scale elliptic operator.
- Using classical Multi-grids and auxiliary-space theory we can perform the invert of the Schur approximation.

Available algorithms

- Tensor decomposition, when using Tensor Spaces
- Metis (ParMetis will be added later)



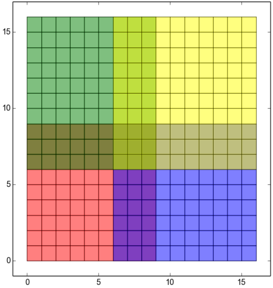
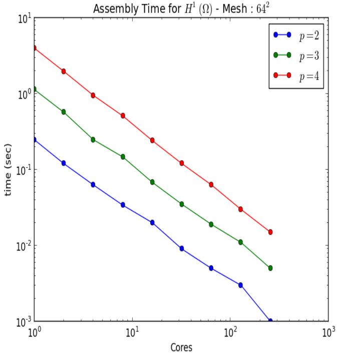
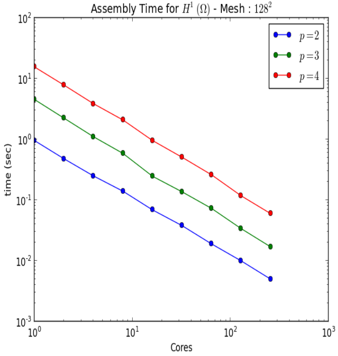




Figure : Metis (left) and tensor (right) partitioning.

# Numerical results: Parallel runs

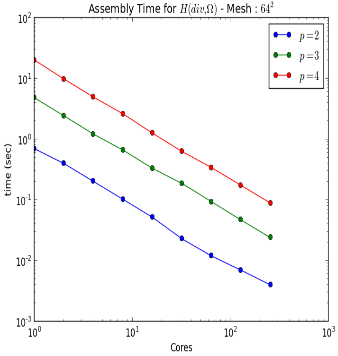
The $2D$ case

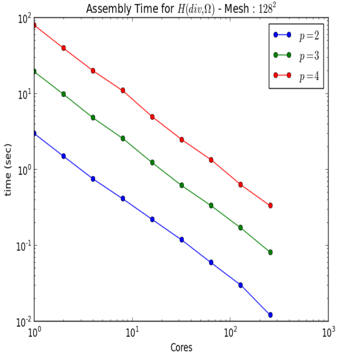


# Numerical results: Parallel runs

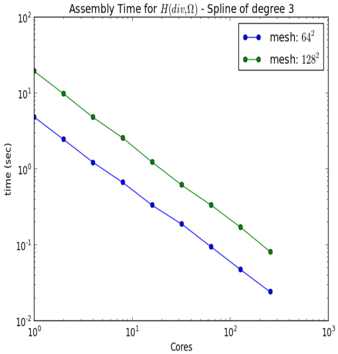
## The $2D$ case

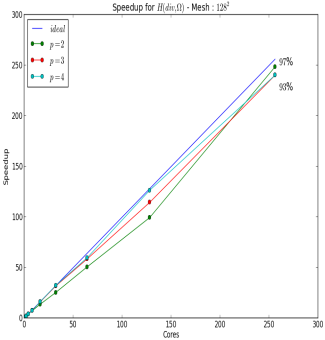


# Numerical results: Parallel runs

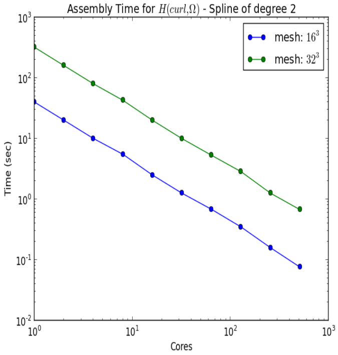
## The $2D$ case

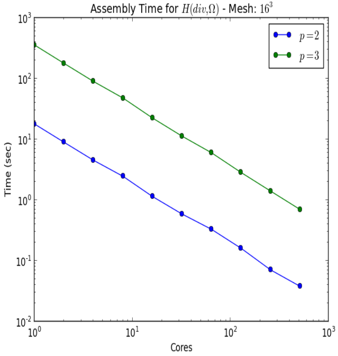


# Application: Parallel runs

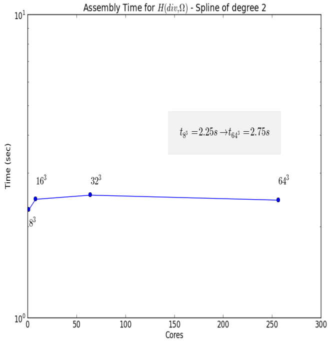
## Parallel assembly for $H(\text{curl}, \Omega)$ and $H(\text{div}, \Omega)$ in 3$D$



# Application: Parallel runs

## Parallel assembly for $H(\text{curl}, \Omega)$ and $H(\text{div}, \Omega)$ in $3D$

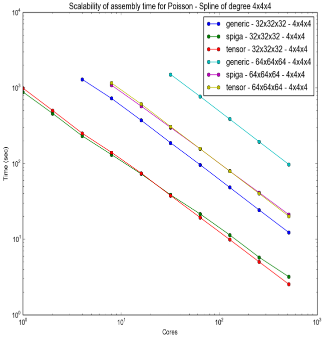



**Statistics:** Quadratic Splines on a grid $32^3$:

- $23'101'440$ non zeros for $H(curl)$
- $98'304$ dofs for $H(curl)$
- $13'860'864$ non zeros for $H(div)$
- $98'304$ dofs for $H(div)$

# Cost of the Object-Oriented implementation

1. How does the use of the procedure pointer for the weak formulation perform compared to the hardcoded version of Poisson?
2. Is there a simple way to enhance and accelerate the assembly procedure taking into account some discretizations properties?



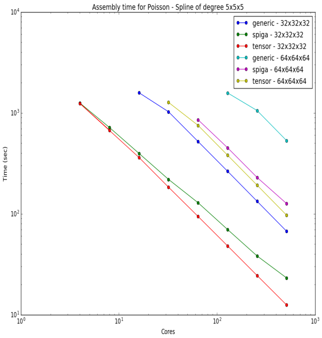




Figure : Scalability of different assembly procedures for quadrtic and quitinc B-Splines.