Journée SFT – 16/06/2025

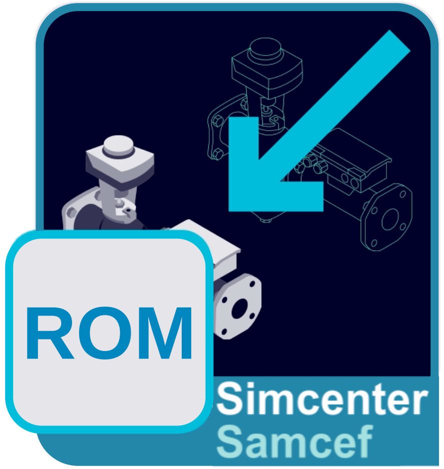
Pierre-Eliot MALLEVAL

SOLVING NONLINEAR AND TRANSIENT PROBLEMS IN AN INDUSTRIAL SOLVER USING A WEAKLY INTRUSIVE MODEL ORDER REDUCTION METHOD

Thesis supervisors:

David NÉRON (Université Paris-Saclay, CentraleSupélec, ENS Paris-Saclay, CNRS, LMPS - Laboratoire de Mécanique Paris-Saclay, 91190, Gif-sur-Yvette, France)

Ronan SCANFF (Siemens Industry Software SAS)



1. CONTEXT

- Model Order Reduction
- Industrial software integration

Context and Motivations – Model order reduction

Reduced-order models (ROM)

- Simplify complex systems while still capturing their essential behavior
- Leverage redundancy of information
- Reduce the complexity to a small number of DOFs

Find $u \equiv u(\mu) \in X$ s.t.

$$\mathcal{L}\left(u, \mu\right) = 0 \quad \forall \mu \in \mathcal{D}$$

$\mathcal{D} \to \mathbb{R}$, $\mathbb{R}^p$, $\dots$

"a parameter" (material parameters, geometric parameter, etc.)

We look for:

$$u\left(\mu_i\right) \to \mathcal{L}(u_i, \mu_i) \to \{u_i\}$$

➔ Reduced-Order Basis

We hope to be able to represent $\mathcal{S}\left(\mathcal{D}\right) = \{u(\mu), \mu \in \mathcal{D}\}$ with a small number of *snapshots*

How should we choose $\mu_i \quad i = 1, \dots, N$ **?**

POD *[Lumley, 1967; Chatterjee 2000]*

(C)RBM *[Maday et al., 2002; Rozza et al. 2007*

Snapshots collection

Reduced basis construction $\boldsymbol{P}$

Offline

Galerkin Projection

$$\boldsymbol{P}^T \mathbf{M} \boldsymbol{P} \dot{\boldsymbol{\theta}}(t) + \boldsymbol{P}^T(t) \mathbf{A} \boldsymbol{P} \boldsymbol{\theta}(t) = \boldsymbol{P}^T \mathbf{Q}_{\text{ext}}\left(t\right)$$

Very fast online resolution

Online

Context and Motivations

Reduced-order models (ROM)

What if we consider time: $\mu = t$?

$\mathcal{D} \equiv \mathbb{R}$ Sampling of a 1D space is easy!

$$\mathcal{L}\,(u, t) = 0 \quad \forall t \in [0, t_{\text{end}}]$$

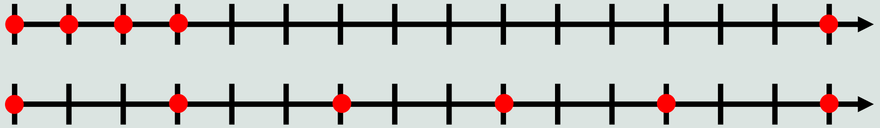
We look for:

$$u\,(t_i) \rightarrow \mathcal{L}(u_i, t_i) \rightarrow \{u_i\}$$

➔ Reduced-Order Basis

Evolution problem ➔ history
Interdependency of the *snapshots*
Definition of time-derivative quantities?

$t_i \quad i = 1, ..., N$



Choice of (computationally expensive) time-snapshots?

The Proper Generalized Decomposition (PGD)

[Ladevèze 85, Chinesta & Ladevèze 14]

- PGD is interested directly in the parametrized solution itself on the **whole time domain:**

$$u(t, M) \quad \forall\,(t, M) \in [0, t_{\text{end}}] \times \Omega$$

- **Based on a separated variable representation:**

$$u(t, M) \approx \sum_{i=1}^{m} \lambda_i(t)\,\mathbf{\Lambda}_i(M)$$

Time function — Space function

- The basis $\{\mathbf{\Lambda}_i(M), \lambda_i(t)\}_{i=1...m}$ is built **on-the-fly** directly from the PDE with a greedy algorithm **without prior knowledge**
- Unlike POD-based methods, the number of spatial problems (of size $NDOFs \gg m$) to solve does not scale on the number of time snapshots N but directly on the number of modes m.

Context and Motivations – Industrial software integration

Integration in industrial software

Implementation of ROM algorithms in an industrial workflow?

- *A posteriori* snapshot-based methods (POD, CRBM)

Rely on standard FEA software for snapshot generation
- ‘certified’ software and data
- benefits from support contracts

[Giraldi et al., 2014; Casenave et al., 2015; Hesthaven et al., 2018; Hammond et al., 2019; Casenave et al., 2020; Vizzaccaro et al., 2020]

- *A priori* PGD (on-the-fly construction of the basis)

Do not rely on classic algorithms for the resolution
- Literature deals with specific application
- Not fully integrated in standard FEA software

[Courard et al.,2016; Zou et al. , 2018; Ghnatios et al., 2021]

Position of presented work

Introduction of PGD to general-purpose commercial FEA software?

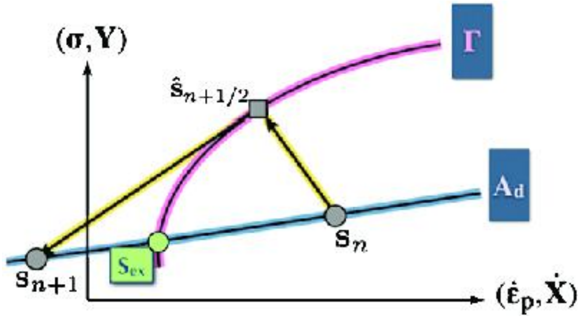
- By leveraging the similarities between Newton-Raphson and LATIN algorithms ***[Scanff et al.,2022]***

Combine ...
- All the sophistication (richness of nonlinear material laws, etc., geometric nonlinearities, element types, etc.),
- robustness and performance

... with …
- A non-incremental algorithm tailored to PGD, *[Ladevèze, 1999]*
- multi-fidelity solver capabilities, *[Nachar et al.,2020]*
- ROM building *[Relun et al., 2013, Heyberger et al., 2013, Scanff et al.,2022, Daby-Seesaram et al.,2025]*

... without altering the (incremental) architecture of the solver

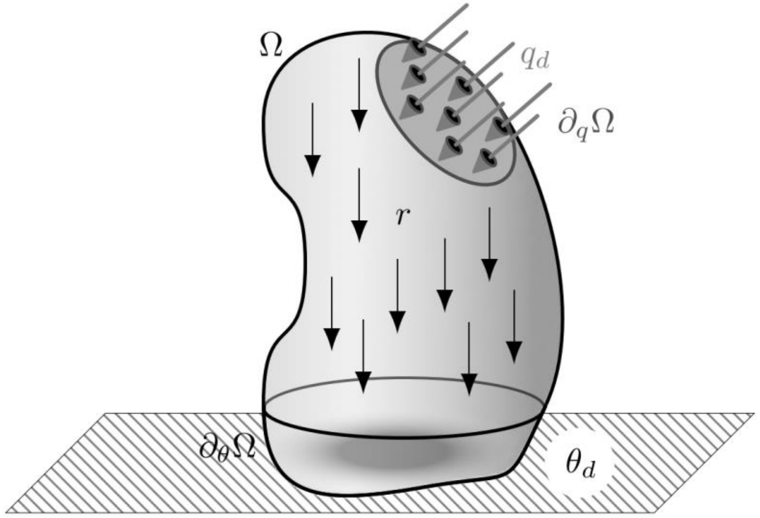
***Weakly intrusive* LATIN-PGD implementation:**
Extension to the transient thermal problem inside the general thermo-mechanical nonlinear FEA solver : **SAMCEF**



2. LATIN-PGD AS AN INDUSTRIAL NON LINEAR SOLVER

- The reference problem
- Weakly intrusive formulation of the LATIN-PGD

The reference problem - heat conduction



Equilibrium equation: $\rho c \frac{\partial \theta}{\partial t} + \nabla \cdot \boldsymbol{q} = r$

Fourier's law: $\boldsymbol{q} = -\boldsymbol{\kappa} \boldsymbol{\nabla} \theta$

Non linear material properties: $\kappa(\theta), \rho(\theta), c(\theta)$

The boundary conditions:
$$\theta = \theta_d \quad \text{over } \partial_\theta\Omega$$
$$-\boldsymbol{q} \cdot \mathbf{n} = q_d \quad \text{over } \partial_q\Omega$$

Including nonlinear convection and radiation BCs

The initial condition: $\theta|_{t=0} = \theta_0$

Solved by any general-purpose commercial finite element software

- Prescribed temperature and initial condition:

$$\boldsymbol{C}_\theta \boldsymbol{\theta}(t) = \theta_d(t) \quad \forall t \in I \quad \text{and} \quad \boldsymbol{\theta}|_{t=0} = \theta_0$$

- Thermal equilibrium between the contribution of the generalized (nodal) heat flows:

$$\mathbf{Q}_{\text{ine}}\left(\dot{\boldsymbol{\theta}}(t), \boldsymbol{\theta}(t); t\right) + \mathbf{Q}_{\text{int}}\left(\boldsymbol{\theta}(t); t\right) - \mathbf{Q}_{\text{ext}}(t) = \mathbf{0} \quad \forall t \in I$$

- Nonlinear heat transfers relationship which result in generalized heat flows:

$$\forall t \in I, \quad \begin{cases} \mathbf{Q}_{\text{ine}}\left(\dot{\boldsymbol{\theta}}(t), \boldsymbol{\theta}(t); t\right) = \mathcal{A}_h(\dot{\boldsymbol{\theta}}(\tau \leqslant t), \boldsymbol{\theta}(\tau \leqslant t); t) \\ \mathbf{Q}_{\text{int}}\left(\boldsymbol{\theta}(t); t\right) = \mathcal{A}_\theta(\boldsymbol{\theta}(\tau \leqslant t); t) \end{cases}$$

$\mathcal{A}_\theta$ $\mathcal{A}_h$ → General form representing the (nonlinear) treatment of local elements by the considered software

Toward the LATIN-PGD weakly intrusive formulation

Newton-Raphson algorithm

Initialization: t_0, θ_0

Time loop: for all time-steps $\{t_p\}$

Nonlinear loop $\blacksquare^{(k)}$: while $\|\mathcal{F}\| \geq \eta$

Local integration of behavior laws: $\mathcal{A}_h$ $\mathcal{A}_\theta$
Linearization of the local behavior
(Assembling of tangent matrix)

$\tilde{\boldsymbol{\Gamma}}_p$

Local equations
Still non-linear

Computation of the correction $\Delta\theta_p^{(k+1)}$
Update of the temperature $\theta_p^{(k+1)}$
Computation of the equilibrium residual $\mathcal{F}$

$\left(\tilde{\boldsymbol{A}}_d\right)_p$

Linear set of equations
Global in space… but
local in time

End of nonlinear loop

End of time loop

Not possible to use space-time PGD in this framework

Weakly intrusive LATIN algorithm

[Ladevèze, 85; Busy et al., 90; Boisse et al., 91; Ladevèze et Perego, 00; Scanff et al., 2022]

Swaps loops between time-steps and convergent iterations:

Initialization: $\theta_0(t), \forall t \in I$ (admissible)

Nonlinear loop $\blacksquare^{(k)}$: while $\eta_{latin} \geq \eta$

Time loop: for all time-steps $\{t_p\}$

Local nonlinear $(\boldsymbol{\Gamma})$

Linear global in space and time

$(\boldsymbol{A}_d)$

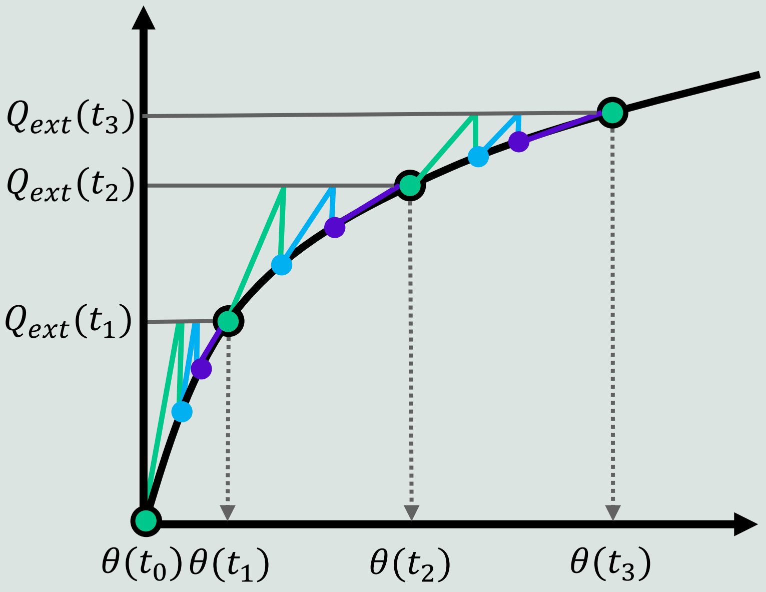
Two-step iterative procedure

End of nonlinear loop

Tailored to PGD

Toward the LATIN-PGD weakly intrusive formulation

Newton-Raphson algorithm

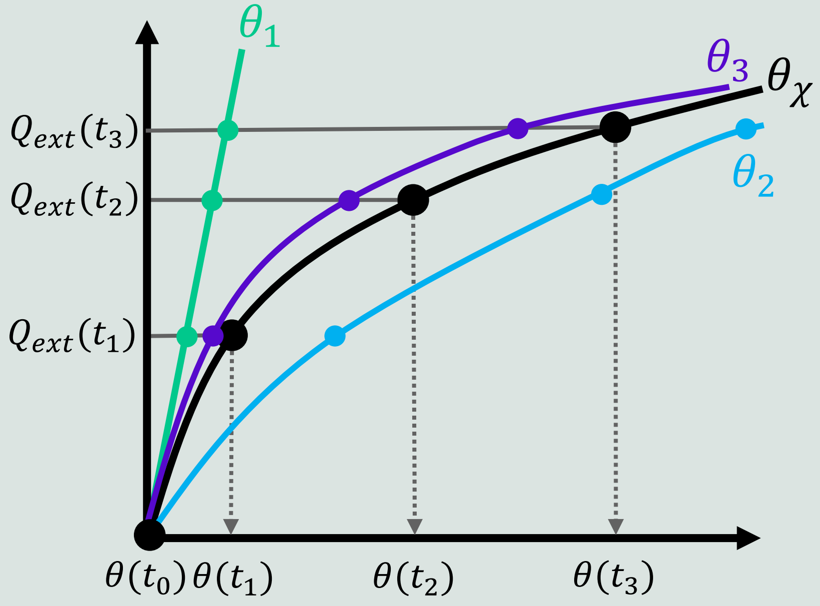


Classical iterative **incremental** approach

- The solution is approximated on **several time-steps** (*"snapshot"*): $u(t_i) = u_i$

Not possible to use PGD in this framework

Weakly intrusive LATIN algorithm

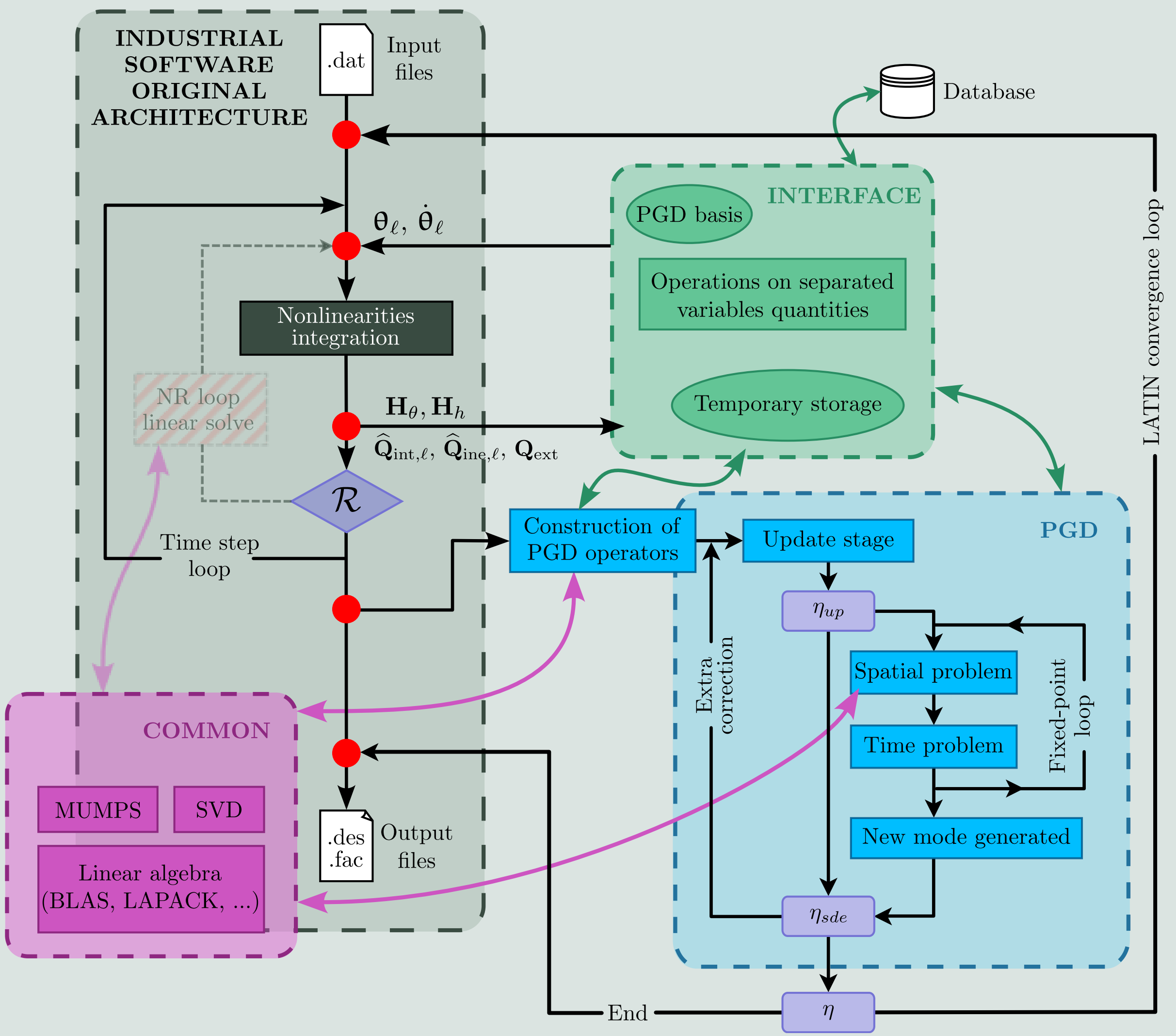


Iterative **Non-incremental** approach

- The solution is approximated on the **whole time interval** at each iteration ℓ

Tailored to PGD: $\boldsymbol{\theta}_\ell(t) = \sum_{i=1}^{m_\ell} \lambda_i(t)\boldsymbol{\Lambda}_i$

Weakly intrusive LATIN-PGD implementation



General architecture of the implementation [Malleval et al.,2025]

$$s_0 \in \boldsymbol{A}_d \dashrightarrow \underbrace{s_\ell \in \boldsymbol{A}_d \xrightarrow{\text{Local stage}} \hat{s}_\ell \in \Gamma \xrightarrow{\text{Global stage}} s_{\ell+1} \in \boldsymbol{A}_d}_{\text{Iteration } \ell+1} \dashrightarrow s_\chi \in \boldsymbol{A}_d \cap \Gamma$$

$$\hat{s}_\ell = \left(\left\{\hat{\theta}_\ell, \widehat{\mathbf{Q}}_{\text{int},\ell}\right\}, \left\{\dot{\hat{\theta}}_\ell, \widehat{\mathbf{Q}}_{\text{ine},\ell}\right\}\right)$$

Constitutive relations $\boldsymbol{\Gamma}$

$$\forall t \in I, \quad \begin{cases} \widehat{\mathbf{Q}}_{\text{ine},\ell}(t) = \mathcal{A}_h(\dot{\hat{\theta}}_\ell(\tau \leqslant t), \hat{\theta}_\ell(\tau \leqslant t); t) \\ \widehat{\mathbf{Q}}_{\text{int},\ell}(t) = \mathcal{A}_\theta(\hat{\theta}_\ell(\tau \leqslant t); t) \end{cases}$$

Ascent search direction $\boldsymbol{\Upsilon}^+$

$$\forall t \in I, \quad \begin{cases} \hat{\theta}_\ell(t) = \theta_\ell(t) \\ \dot{\hat{\theta}}_\ell(t) = \dot{\theta}_\ell(t) \end{cases}$$

Computed with the solver capabilities within the time step loop

$$s_\ell = \left(\left\{\theta_\ell, \mathbf{Q}_{\text{int},\ell}\right\}, \left\{\dot{\theta}_\ell, \mathbf{Q}_{\text{ine},\ell}\right\}\right)$$

Admissibility and equilibrium equations $\boldsymbol{A}_d$

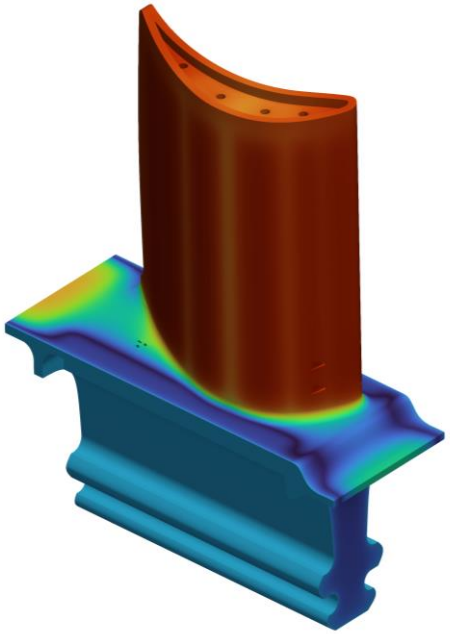
$$\forall t \in I, \quad \mathbf{Q}_{\text{ine},\ell+1}(t) + \mathbf{Q}_{\text{int},\ell+1}(t) - \mathbf{Q}_{\text{ext}}(t) = \mathbf{0}$$

Descent search direction $\boldsymbol{\Upsilon}^-$

$$\forall t \in I, \quad \begin{bmatrix} \mathbf{H}_\theta(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_h(t) \end{bmatrix} \begin{bmatrix} \theta_{\ell+1}(t) - \hat{\theta}_\ell(t) \\ \dot{\theta}_{\ell+1}(t) - \dot{\hat{\theta}}_\ell(t) \end{bmatrix} = \begin{bmatrix} \mathbf{Q}_{\text{int},\ell+1}(t) - \widehat{\mathbf{Q}}_{\text{int},\ell}(t) \\ \mathbf{Q}_{\text{ine},\ell+1}(t) - \widehat{\mathbf{Q}}_{\text{ine},\ell}(t) \end{bmatrix}$$

$\mathbf{H}_\theta(t)$ $\mathbf{H}_h(t)$ Search directions computed from standard FE operators

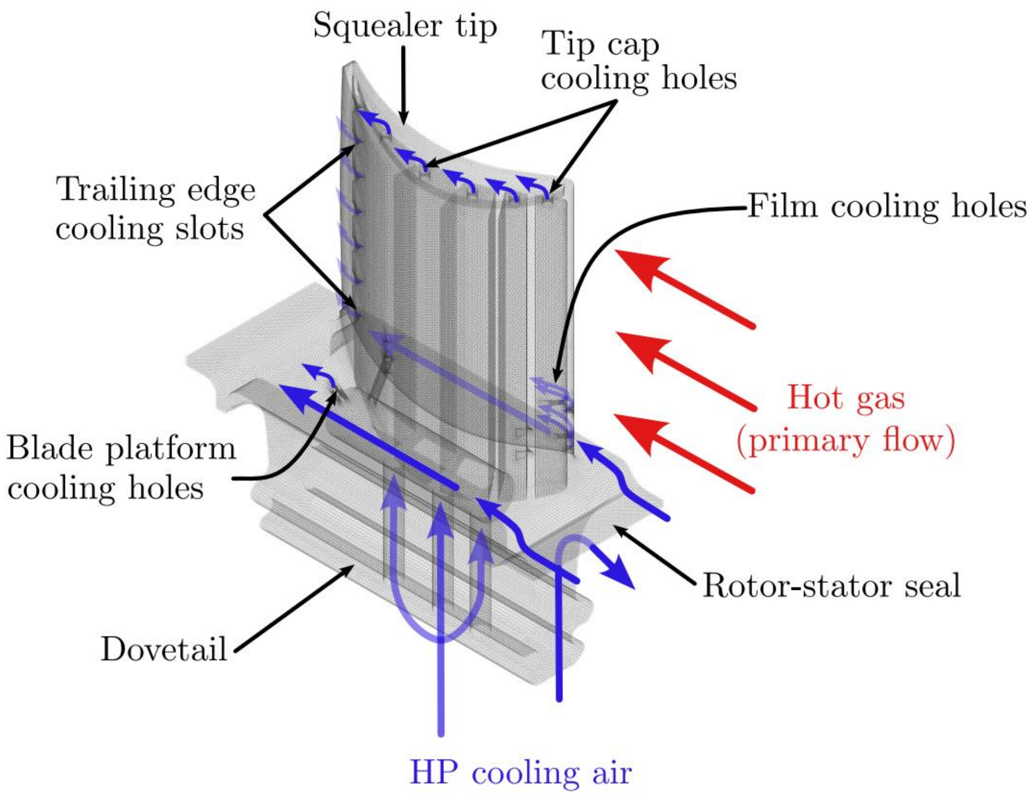
Linear problem solved with PGD at each LATIN iteration



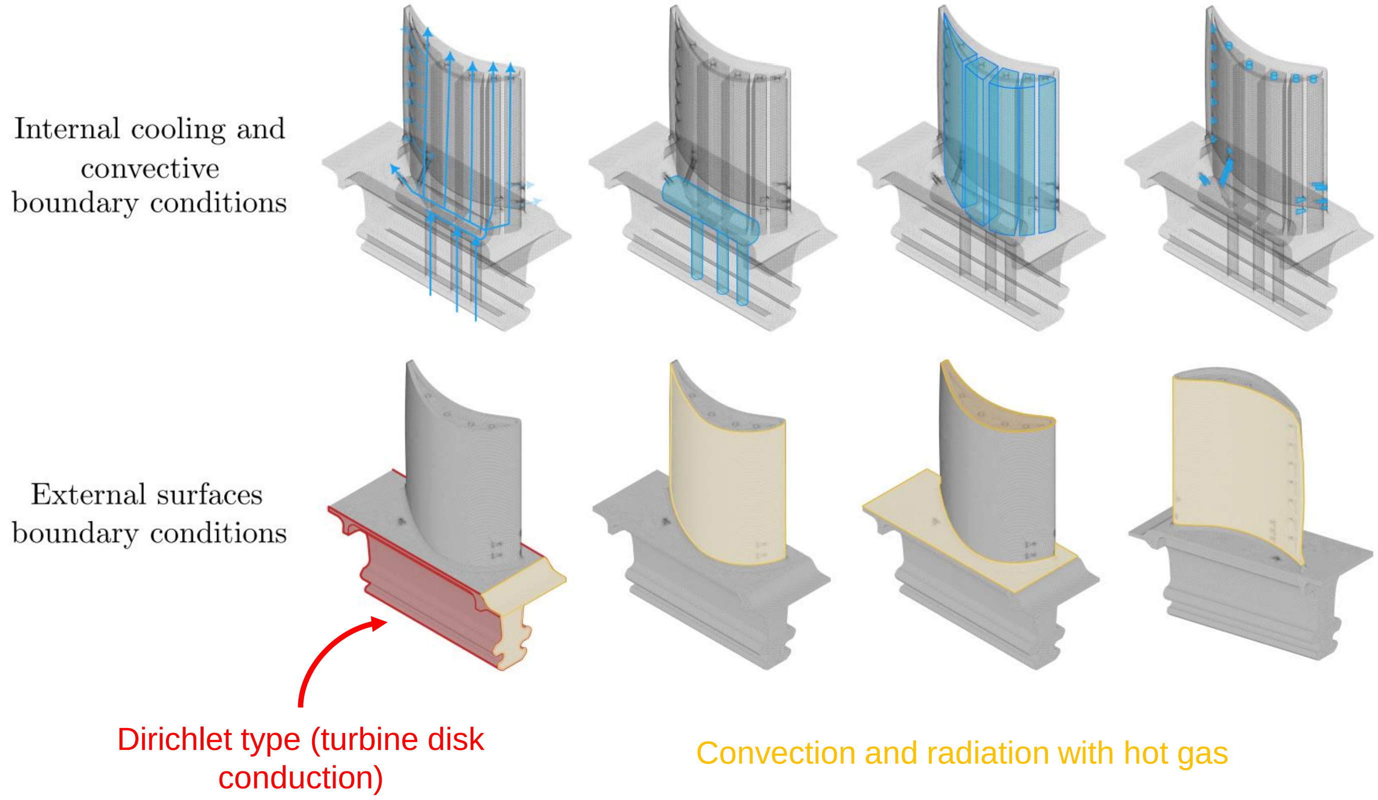
3. Some Results

- Turbine blade thermal test-case
- Computational performance

Turbine blade thermal test-case



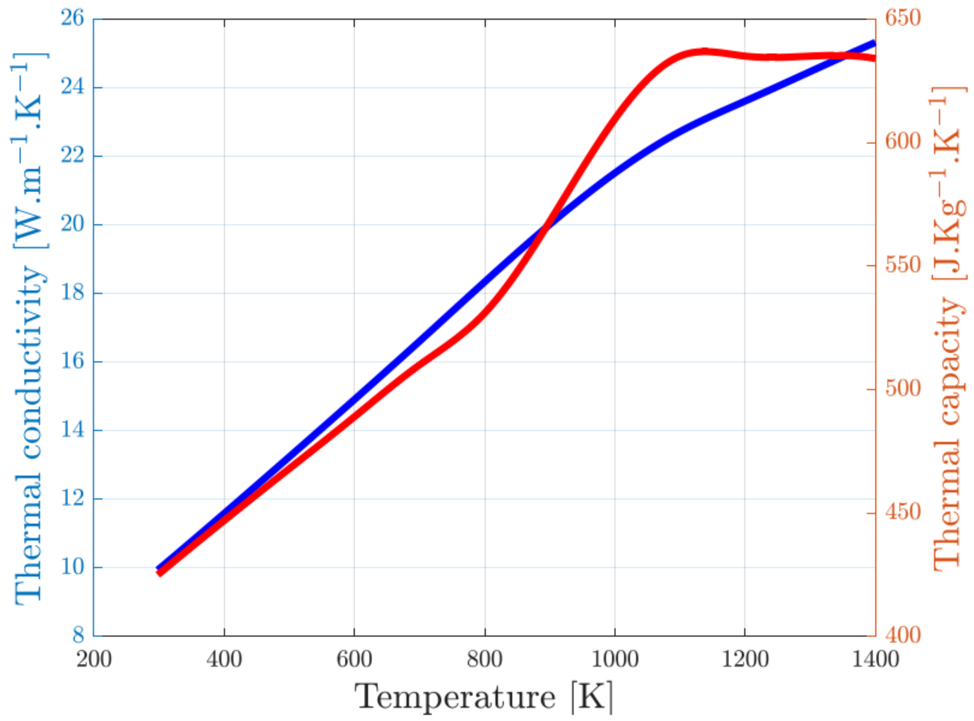
Gas turbine blade cooling schematic



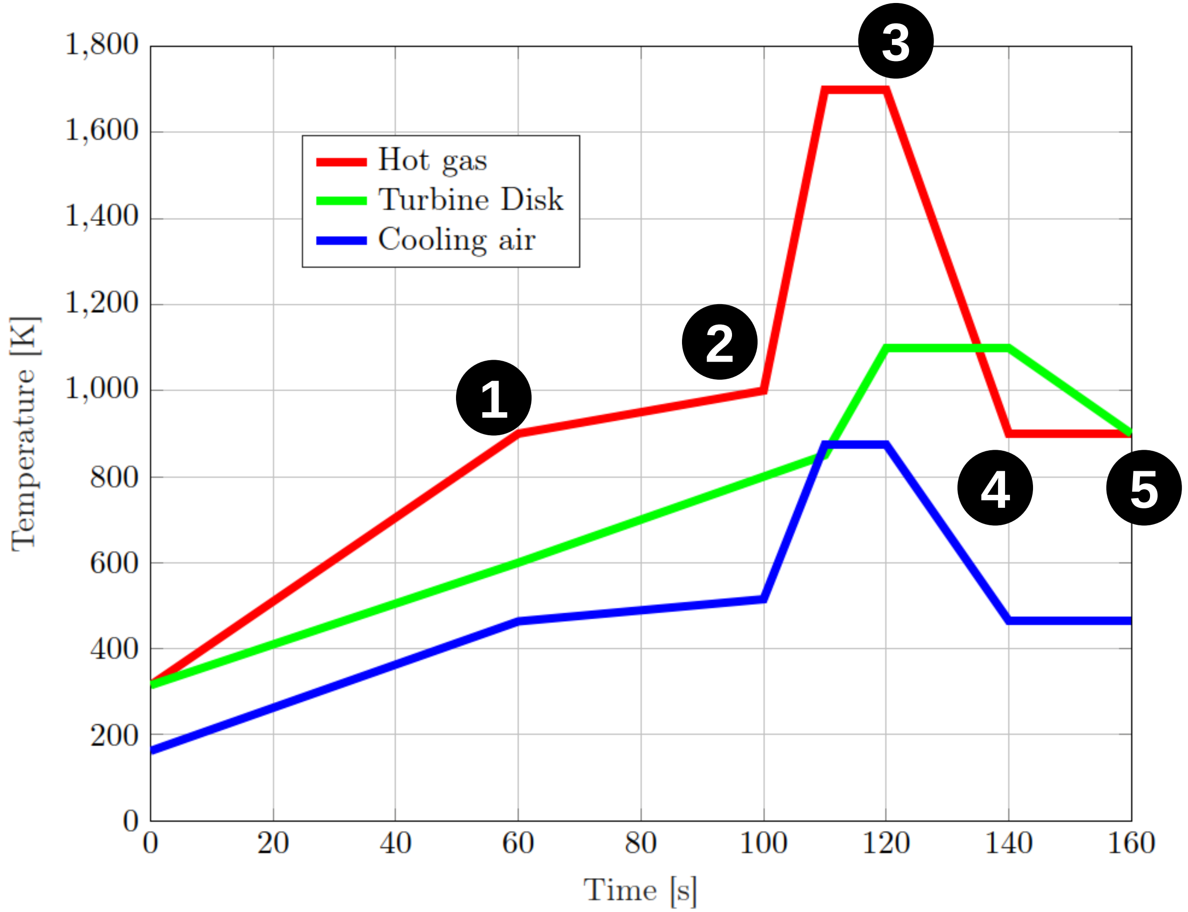
Representation of the boundary conditions applied in Simcenter3D

➢ The test-case is entirely set up in Simcenter3D software (mesh of ~1m6 dofs)

Turbine blade thermal test-case



Material parameters (Inconel 718 superalloy from [Pottlacher,02])

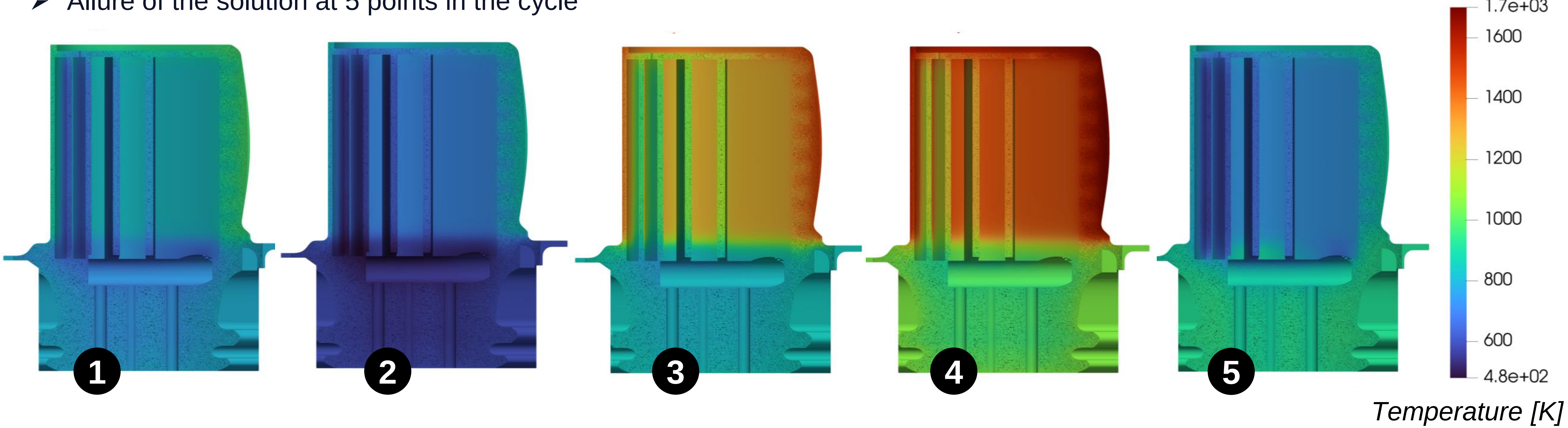


Imaginary loading scenario ($\theta_{max} = 1700K$)

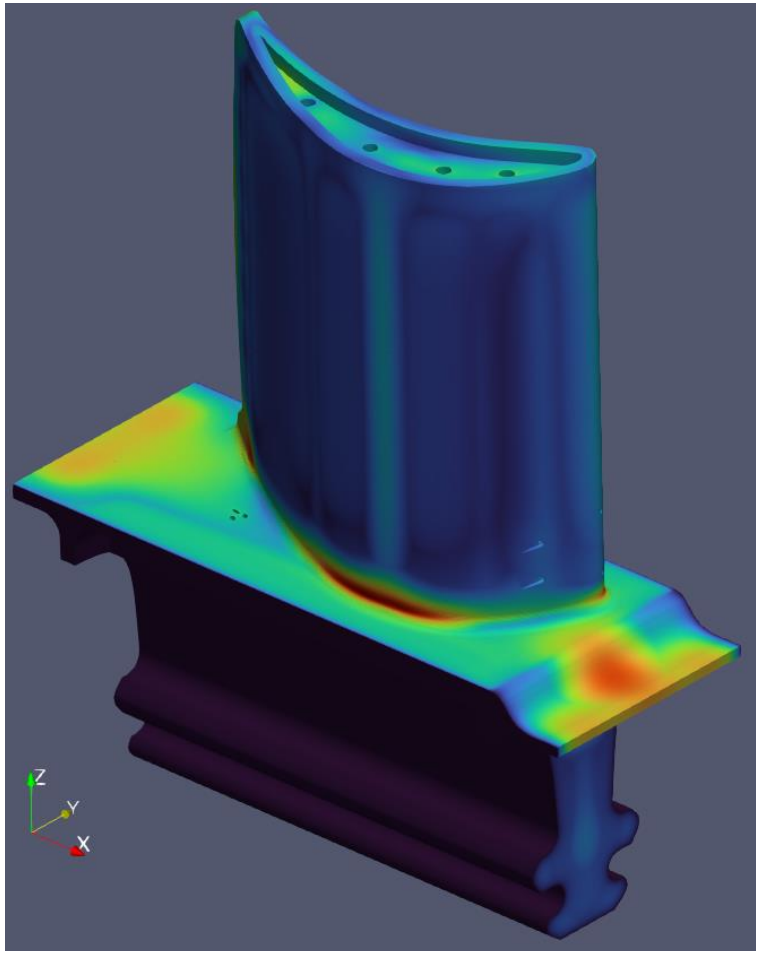
➢ The results computed with the **LATIN-PGD in Simcenter Samcef** are compared to a reference solution obtained in Simcenter Samcef with the Newton-Raphson method.

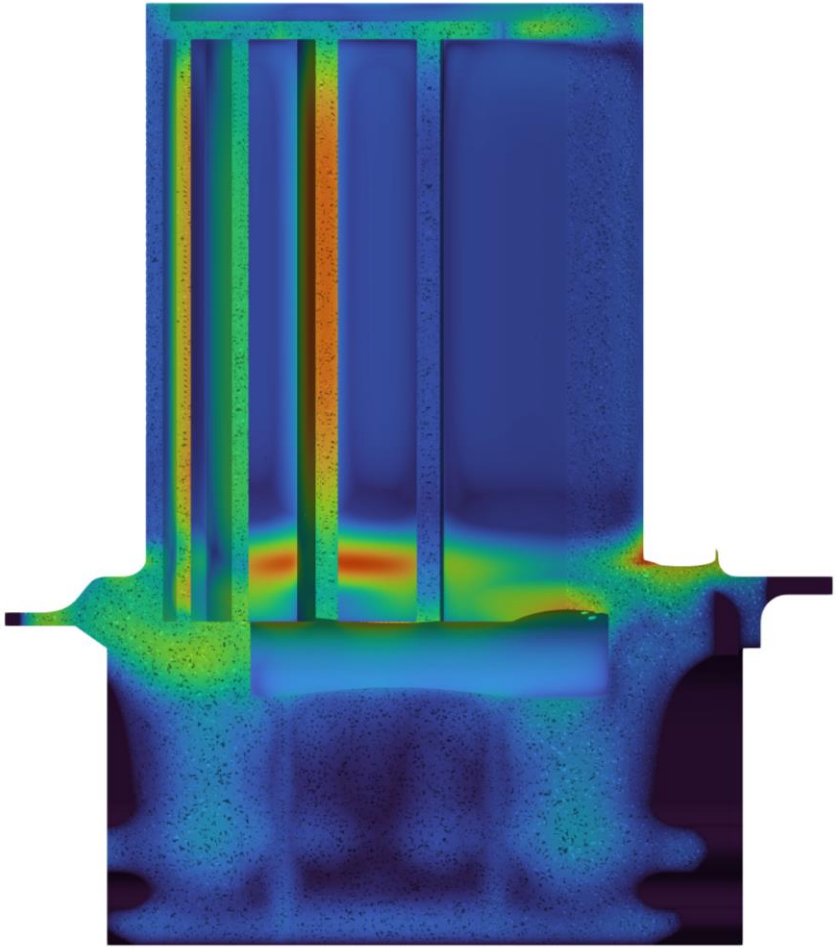
Turbine blade thermal test-case

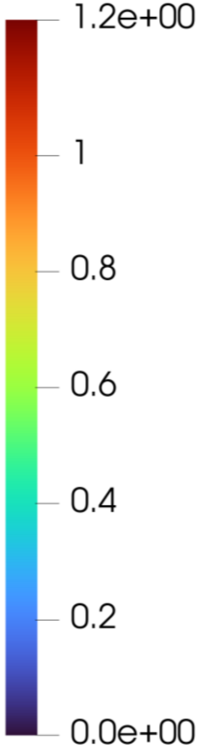
- Allure of the solution at 5 points in the cycle



Turbine blade thermal test-case





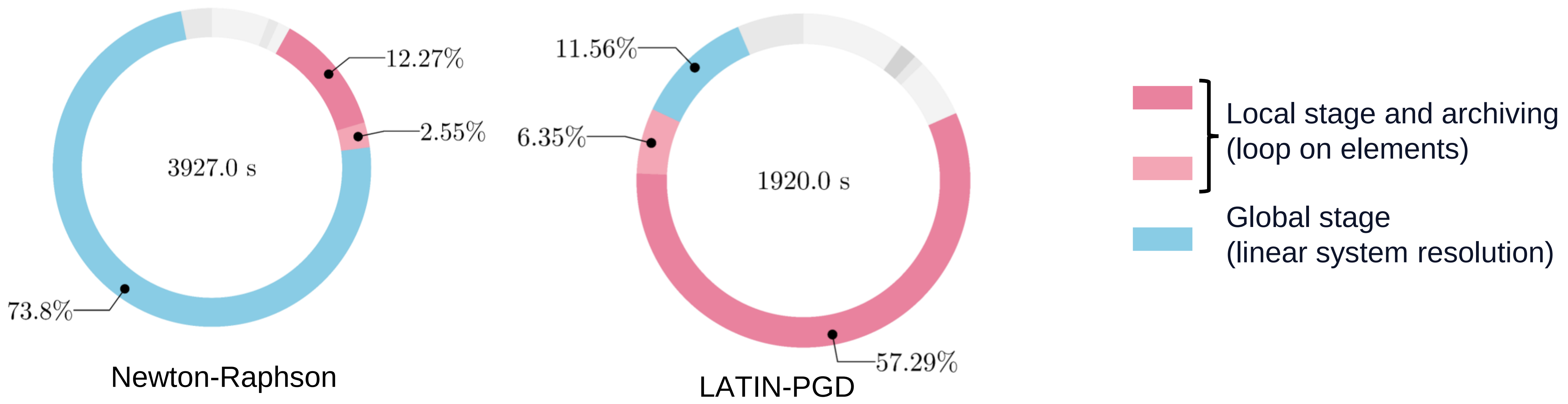


True error at convergence,
computed as: $\max_{t\in I} 100 \frac{|\theta_l(t)-\theta_\chi(t)|}{|\theta_\chi(t)|}$

- At iteration 10 (b) : The maximal error is 1.2%; it is localized on the upper part of the stator-rotor seal and on the internal cooling cavities wall. The outside surface of the blade subjected to high temperature is close to 0.2% error.

Computational performance : profiling

Reference Newton-Raphson compared to the LATIN-PGD



Computation time
(10 cores Intel Xeon Gold 6346 CPU @ 3.10 GHz 36.864 MB L3 cache)

- Total computation time is divided by ~2.
- The CPU time allocated to linear system resolution is divided by 13 while the number of behavior integration is increased by a factor 2.1
- The obtained reduced basis can be re-used to speed up further computation

Computational performance : the local stage bottleneck

Poor performance of the local stage

Algorithm

- The functional formulation of the LATIN reduces performance of the local stage

Implementation

- Performance of the Newton-Raphson local stage, which is not as optimized as linear system resolution by developers of the original software
- Developments choices related to the minimally intrusive framework

Optimization of the local stage

To reduce the cost of the local stage, it is possible to work on:

- ❑ The number of LATIN iteration n_l
  Optimization of the reduced basis computation and search directions
  [PE Malleval, R Scanff, D Néron, 2025]
- ❑ The number of time-steps n_t
  Adaptative time-stepping strategy : adapt features of the original code for LATIN-PGD
- ❑ Directly the number of DOFs in the elements loop
  Hyper-reduction : aDEIM? ➔ challenge of adaptivity and high development cost in Samcef
  [Peherstorfer, B., and Willcox, K, 2015]
- ❑ The evaluation of nonlinear terms
  Replace local integration with trained ANN
  → need to target specific material / model
  [PE Malleval, V Matray, et al., 2025. Pre-print ⟨hal-05070128⟩]

5. CONCLUSION

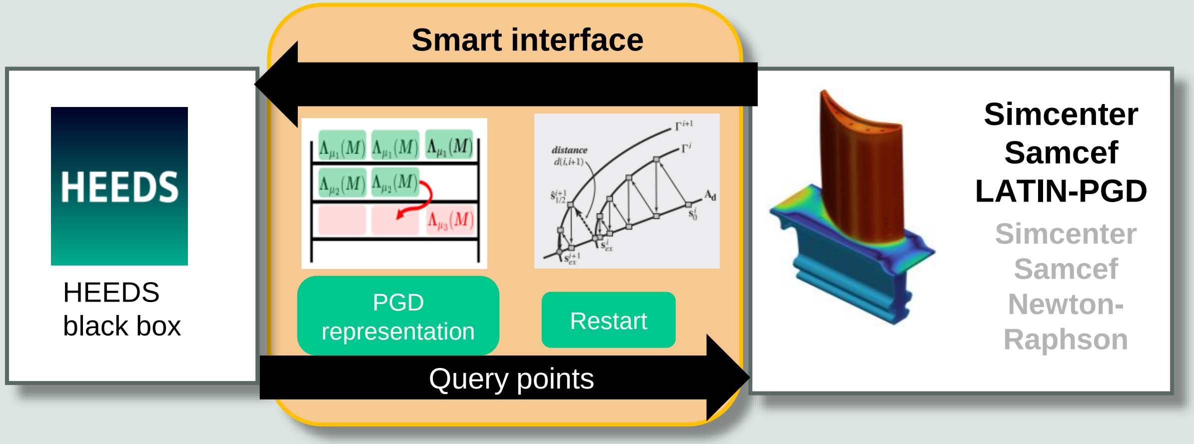
Conclusion

Conclusion

- Successful **implementation of the PGD inside Simcenter SAMCEF** general purpose FEA software
- **Minimally intrusive implementation** carried out with the use of the LATIN-PGD algorithm (generalized formulation)
- **Completely transparent for the end-user** for any nonlinear thermo/mechanical computation: by switching the LATIN-PGD parameter on (instead of Newton-Raphson) in the input file
- First steps to tackle the “local stage” bottleneck in nonlinear ROM (also interesting results for the FOM)

Perspectives (WIP)

- Assess performance of the ROM in multi-query context instead of single computation
- **Coupling** of the time-space LATIN-PGD nonlinear solver **with a parametric space exploration tool**: HEEDS
- Leveraging the specific features of LATIN-PGD: computation of a reduced basis on-the-fly and ability to restart from any previous admissible field



THANK YOU!