Solvendo est dominium

P. Gosselet

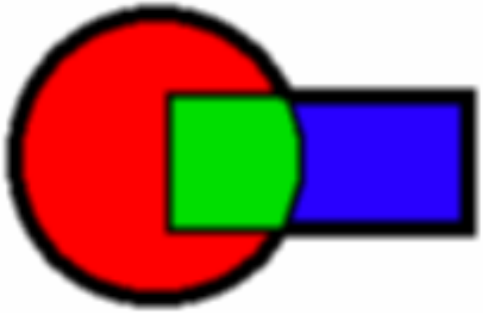
gosselet@lmt.ens-cachan.fr
LMT-Cachan – ENS Cachan/CNRS/Université Paris-Saclay

50th anniversary of the chair of Applied Mechanics
TU-Munich

The domain must be decomposed

A bit of history

- Domain decomposition methods were initiated by Hermann Schwarz circa 1870 in order to prove the existence of solutions to the Laplace equation for domains constituted by the union of overlapping subdomains of simple shape.



  The idea was to build a sequence of solutions on each subdomain such that the difference in the overlap tends to zero thanks to **contractive fixed point** iterations.

- In the absence of overlap, the fields must be **continuous** at the interface and the fluxes must be **balanced** (action-reaction principle). But it is a bit more complex to make stationary iterations converge.

- Then the theory of PDEs improved (Sobolev spaces and variational approaches), and so did the linear solvers (Krylov solvers) and domain decomposition methods are now a powerful framework for modeling, computing,...

1. The principles of Domain Decomposition Methods
2. Handling nonlinearity with DD
3. Verification with DD
4. Modeling with DD

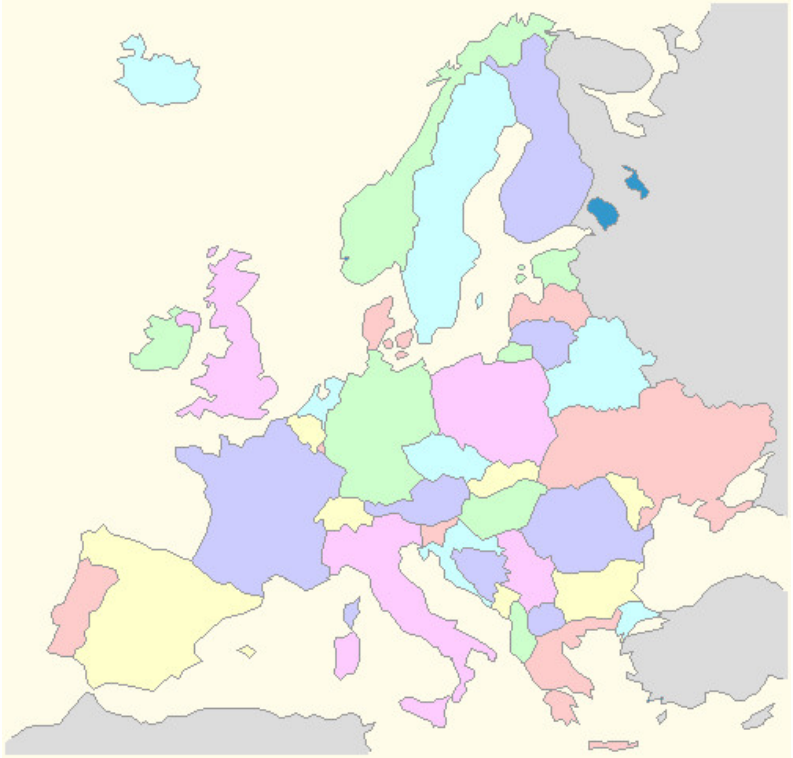
Principle of DD

(1) Take a domain

Principle of DD

(1) Take a domain

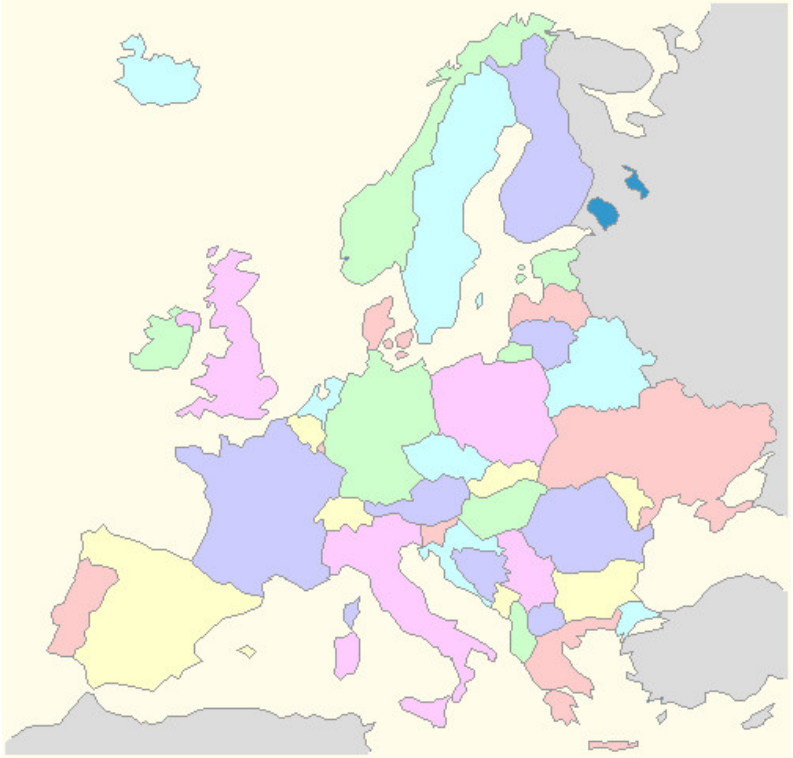
(2) Split it



Principle of DD

(1) Take a domain

(2) Split it

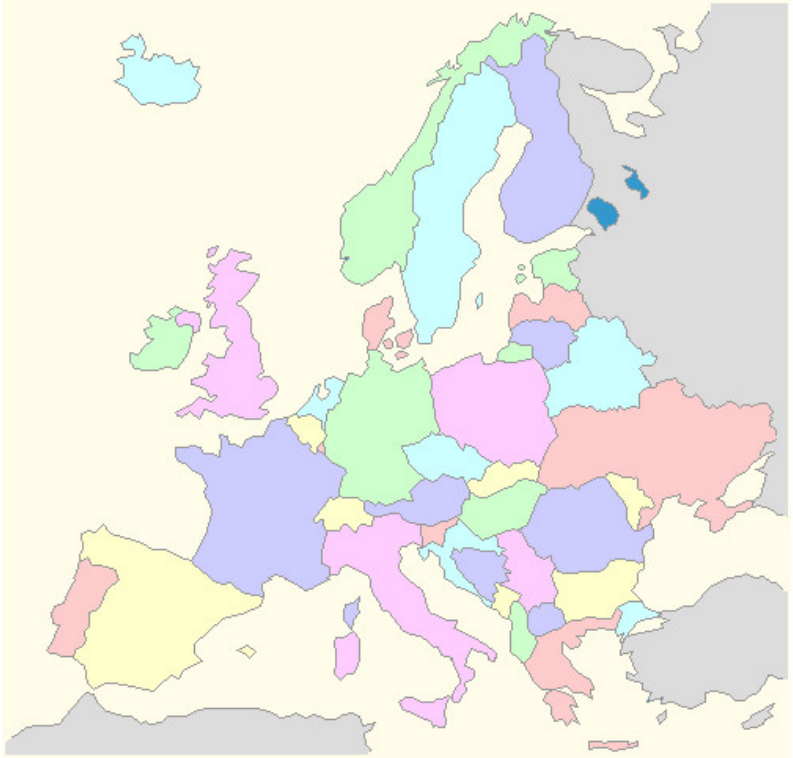


(3) Assume each subdomain is able to rule itself correctly
(in independent processors)

Principle of DD

(1) Take a domain

(2) Split it



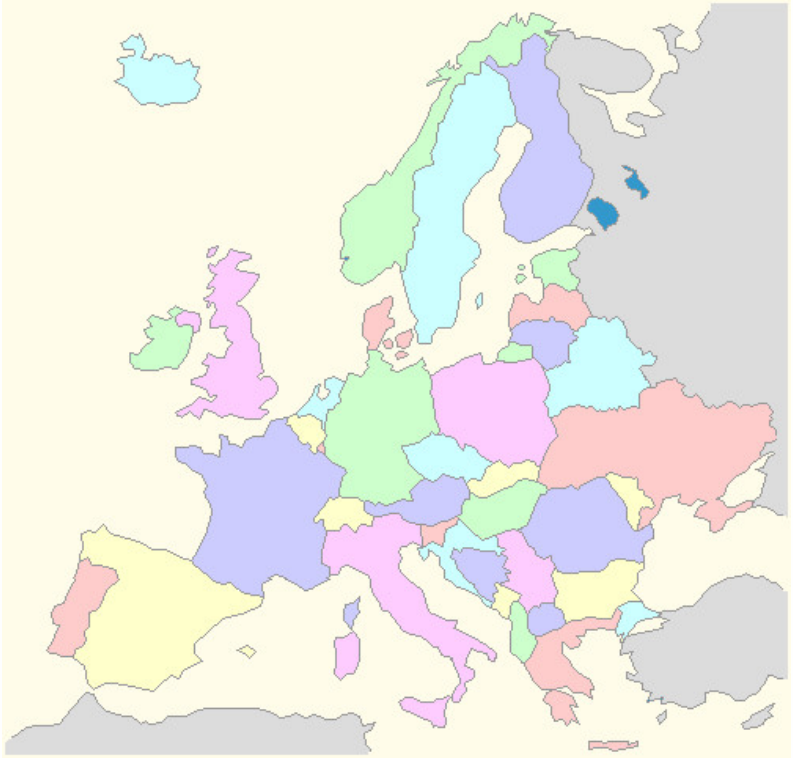
(3) Assume each subdomain is able to rule itself correctly
(in independent processors)

(4) Deal with interactions

Principle of DD

(1) Take a domain

(2) Split it

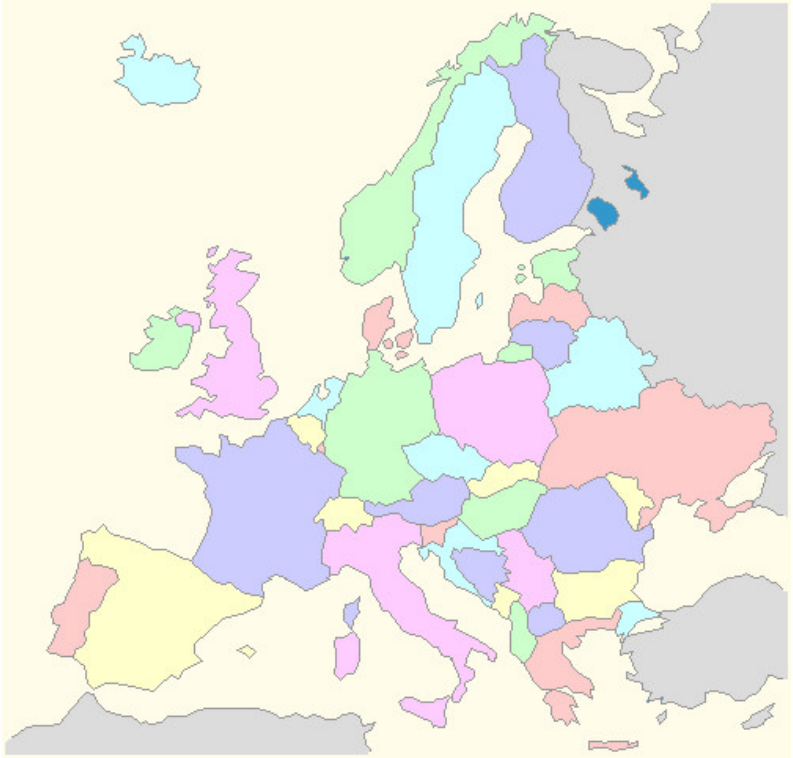


(3) Assume each subdomain is able to rule itself correctly
(in independent processors)
→ Have a good **local solver**
(4) Deal with interactions
→ Make **iterations !**

Principle of DD

(1) Take a domain

(2) Split it



(3) Assume each subdomain is able to rule itself correctly
(in independent processors)
→ Have a good **local solver**
(4) Deal with interactions
→ Make **iterations !**
In mechanics interactions are local, that is to say only through the **boundaries**.
Anyhow local phenomena can have **long-range** effects.

The ingredients of DD methods

Know your neighbors well

The solution is strongly influenced by the properties of the **neighbors**.
One subdomain needs to have a good knowledge of its neighbors through well chosen **boundary conditions**, or well built **preconditioners**.

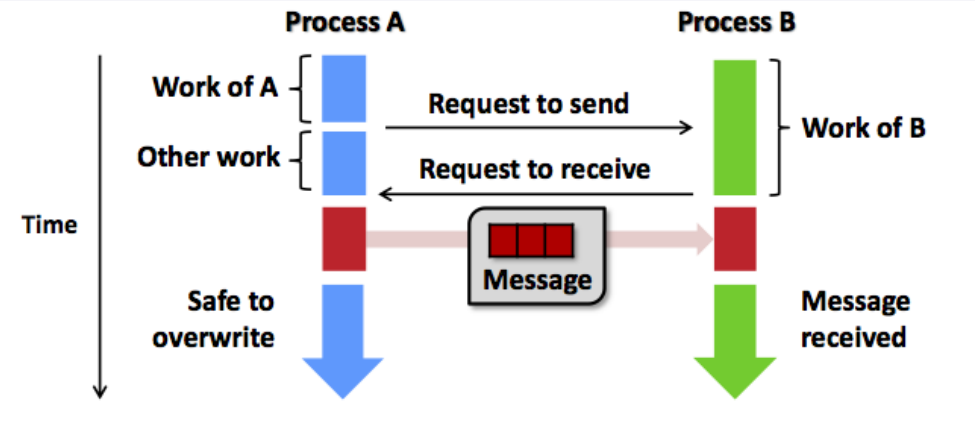
Anticipate long-range effects

Basically information goes through one subdomain by iteration.
This means that the number of iterations explodes with the number of subdomains.
Saint-Venant's principle enables to tell a priori what the **long-range** effects will be.

Message passing point of view

Distributed memory

- Distributed memory is the classical model for large scale computations.
- Each processor has exclusive access to its memory where its subdomain data is.
- Exchanges must be scheduled in order to communicate. The basic operation is send/receive between two processors



http://www.skirt.ugent.be/skirt/_parallelization_m_p_i.html

Latency is a problem so that an objective is to limit communications. Also synchronism is appreciated (data must be available at the right moment), load must be balanced between subdomains.

In reality

Architectures are hybrid and hierarchical, codes must adapt with the help of high level optimized libraries.

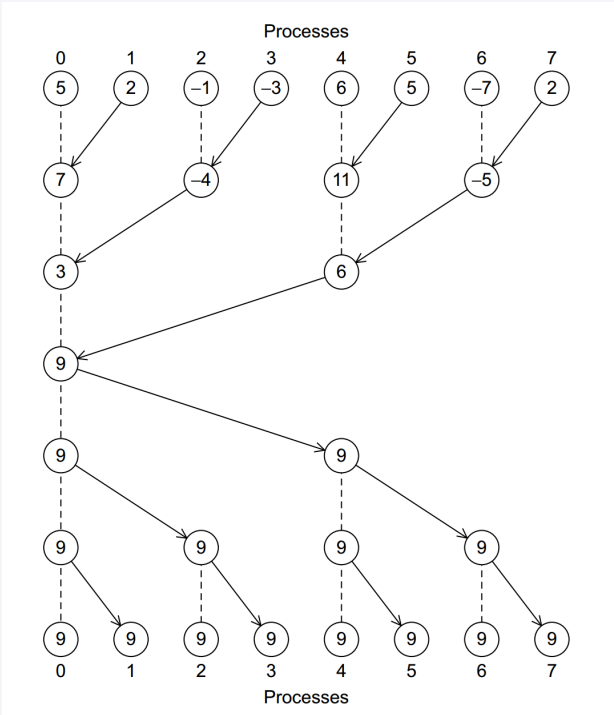
Two categories of communications

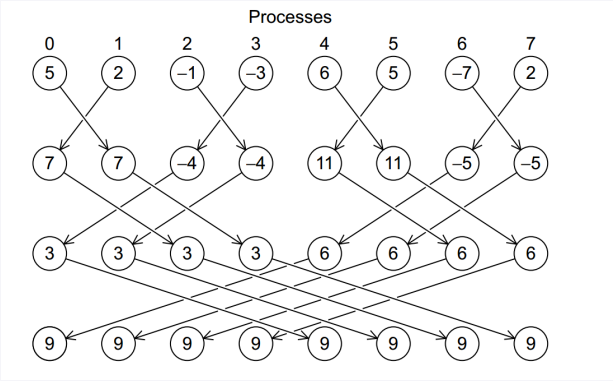
Neighbors communication

It is necessary to exchange data (displacements, nodal reactions) with neighbors.
Often all subdomains must communicate with all their neighbors at the same time.
This is known as a sparse operation.

Collective commmunications

In order to spread the long-range information and to optimize the solving (using dot products in Krylov solvers), it is required to have all-to-all reductions.
A typical operation is : each subdomain owns a double, you want all the subdomains to known the sum.





http://cs.umw.edu/ finlayson/class/fall16/cpsc425/notes/16-collective.html

Local interactions

Basic subdomain solves

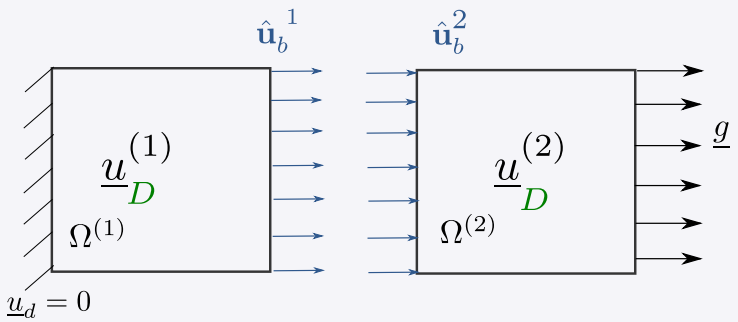
Continuous displacements on the interface

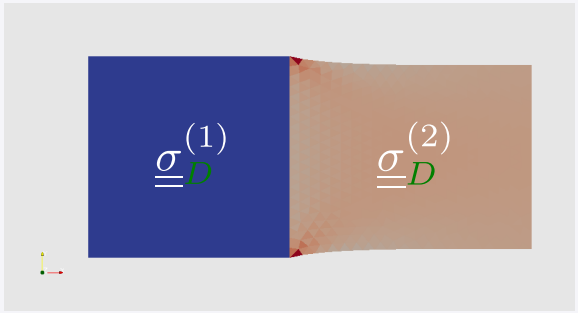
$$\hat{\mathbf{u}}_b^2 - \hat{\mathbf{u}}_b^1 = 0 \Rightarrow \hat{\mathbf{u}}_b^2 = \hat{\mathbf{u}}_b^1$$

Balanced reactions on the interface

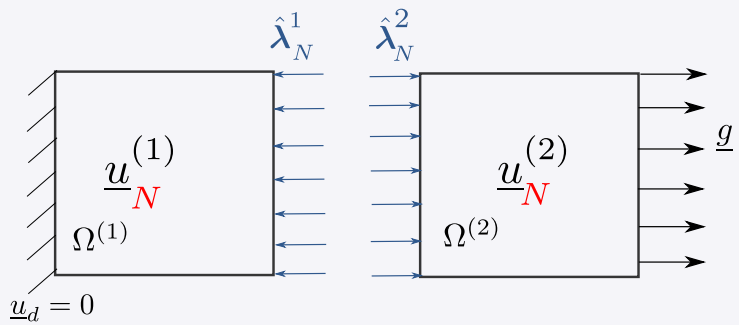
$$\hat{\boldsymbol{\lambda}}_N^2 + \hat{\boldsymbol{\lambda}}_N^1 = 0 \Rightarrow \hat{\boldsymbol{\lambda}}_N^2 = -\hat{\boldsymbol{\lambda}}_N^1$$

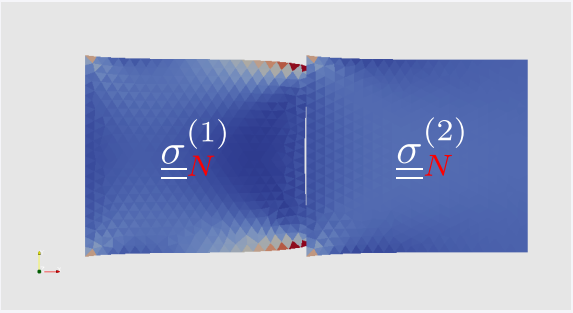
Resolution of Dirichlet problem on $\Omega^{(s)}$





Resolution of Neumann problem on $\Omega^{(s)}$

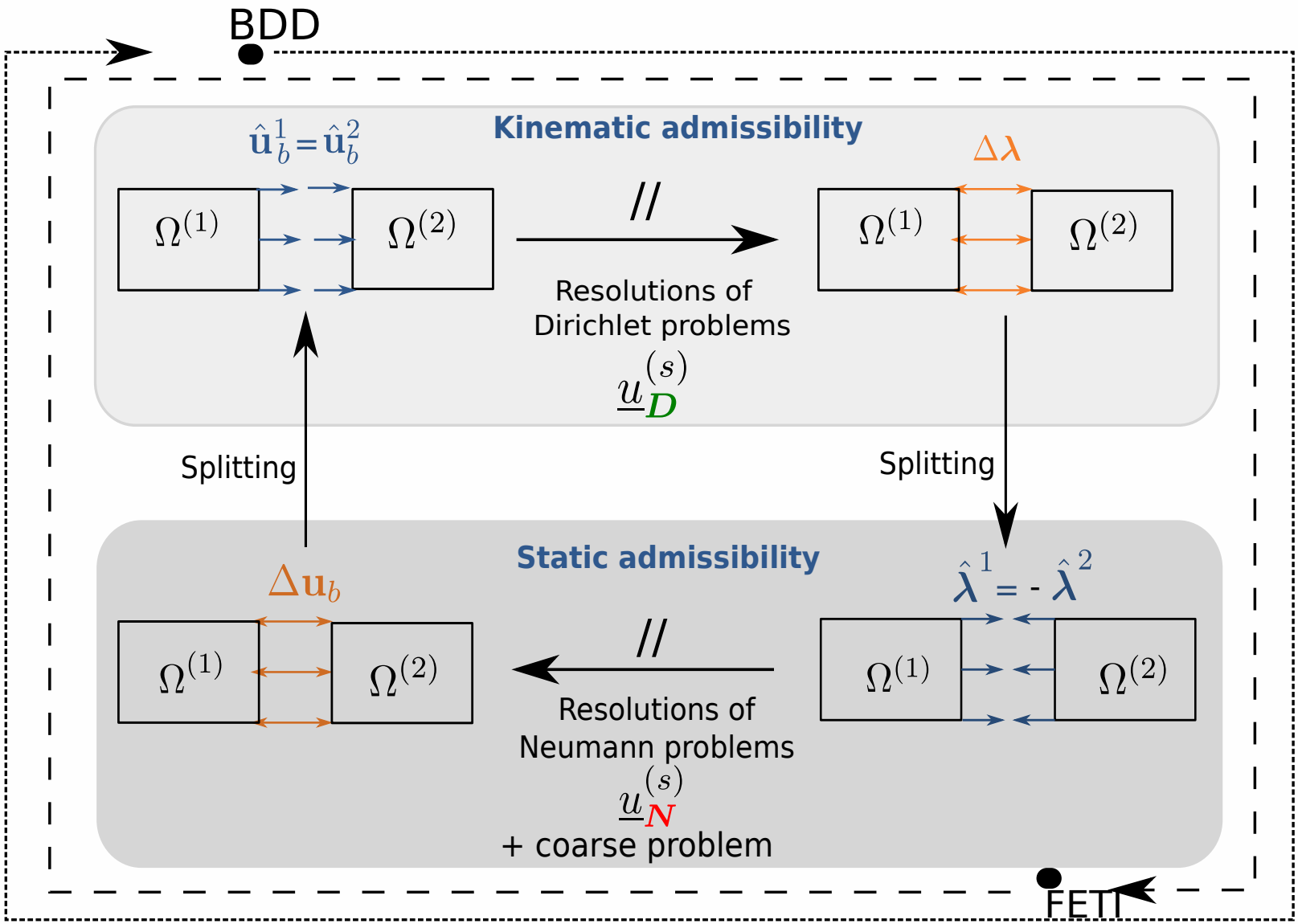




A better representation of the neighborhood than Dirichlet or Neumann bcs can be achieved using (generalized) Robin conditions.

Typical algorithm with two subdomains

Dual approach (FETI), primal approach (BDD)

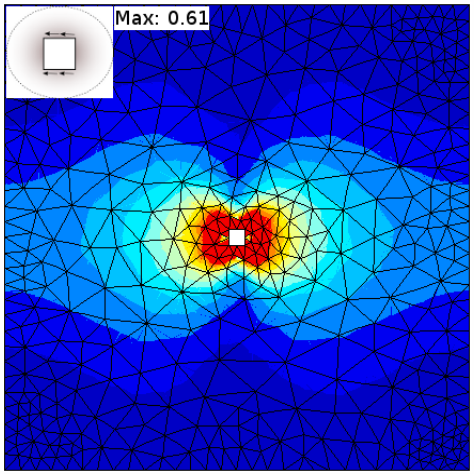


+ **In that case stationary iteration does not work so Krylov solver is mandatory**

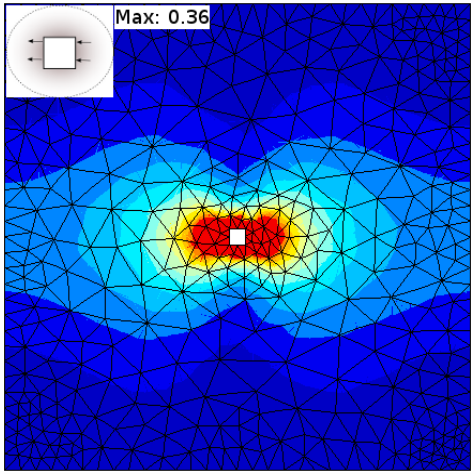
Long range effects, Saint-Venant's principle

Scale separation

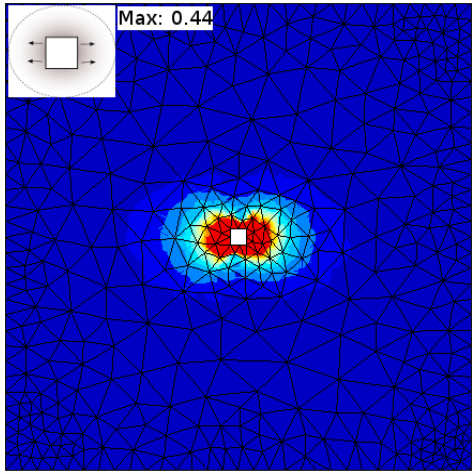
We study the effect of given distributions of traction at the boundary of one subdomain.



Resultant $R_1 = R_2$



Resultant $R_2 = R_1$



Resultant $R_3 = 0$

- Tractions with identical resultants have similar long-range effects.
- Tractions with null resultant only have localized effects.
- Null resultants identify with the orthogonal to rigid body motions.

So we know what should be globally transfered, to do so we use what we call a coarse grid.
The analysis is not as straightforward for complex cases (heterogeneities, subdomains with bad shapes) but theory is now clear (cf Geneo coarse space).

Saint-Venant principle is not enough ...

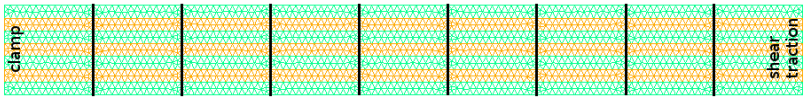


Figure: Beam with straight decomposition

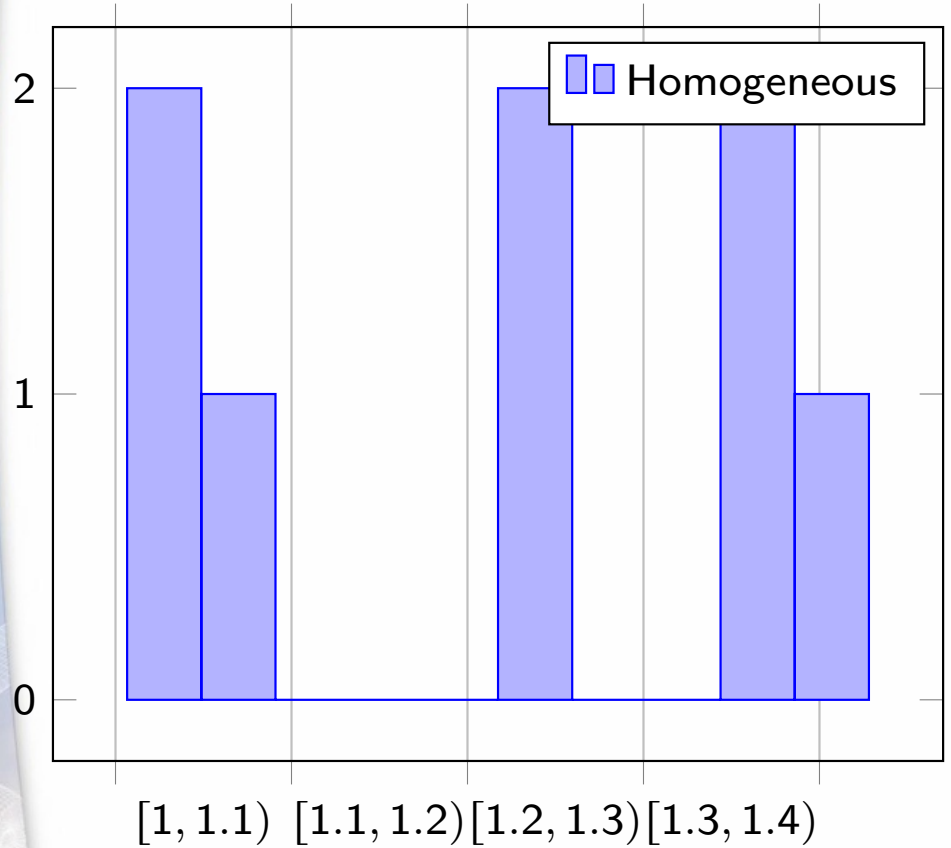


Figure: Spectrum

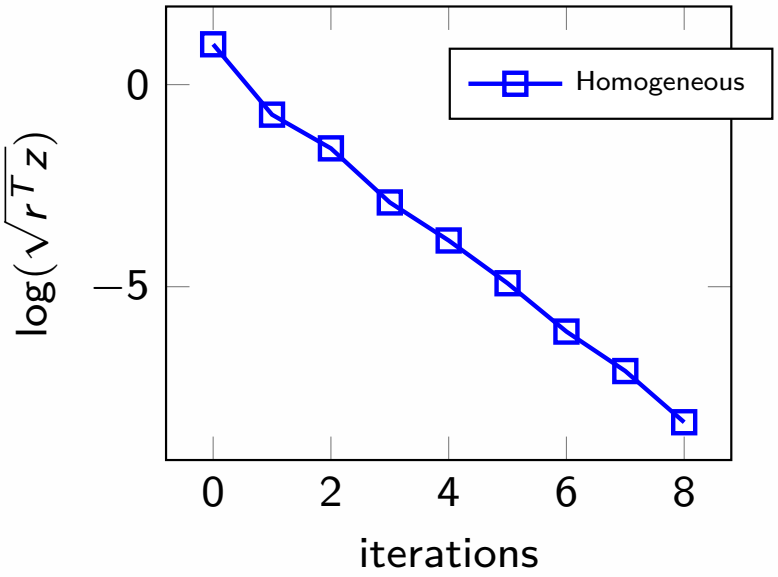


Figure: Convergence

Saint-Venant principle is not enough ...

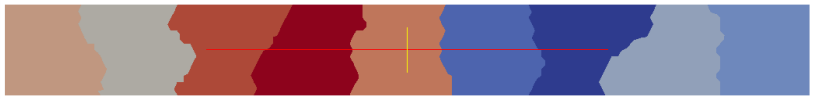


Figure: Beam with irregular decomposition

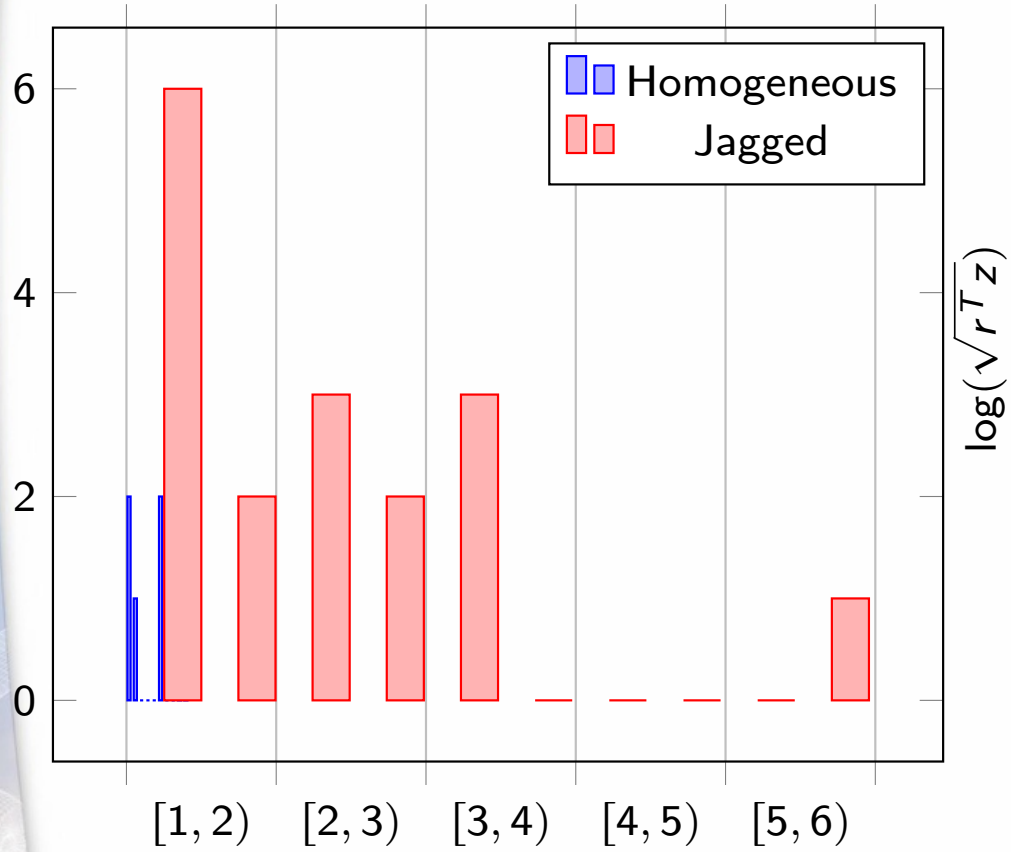


Figure: Spectrum

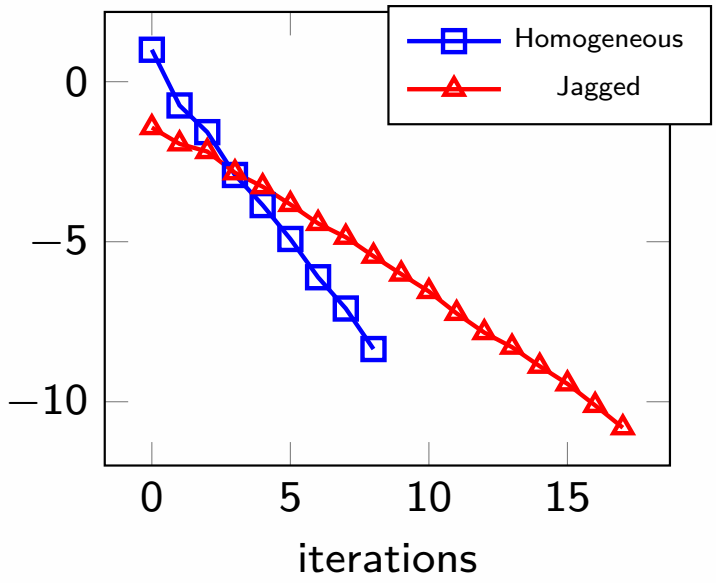


Figure: Convergence

Saint-Venant principle is not enough ...

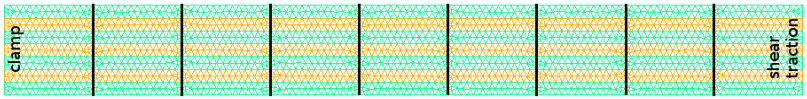


Figure: Beam with straight decomposition

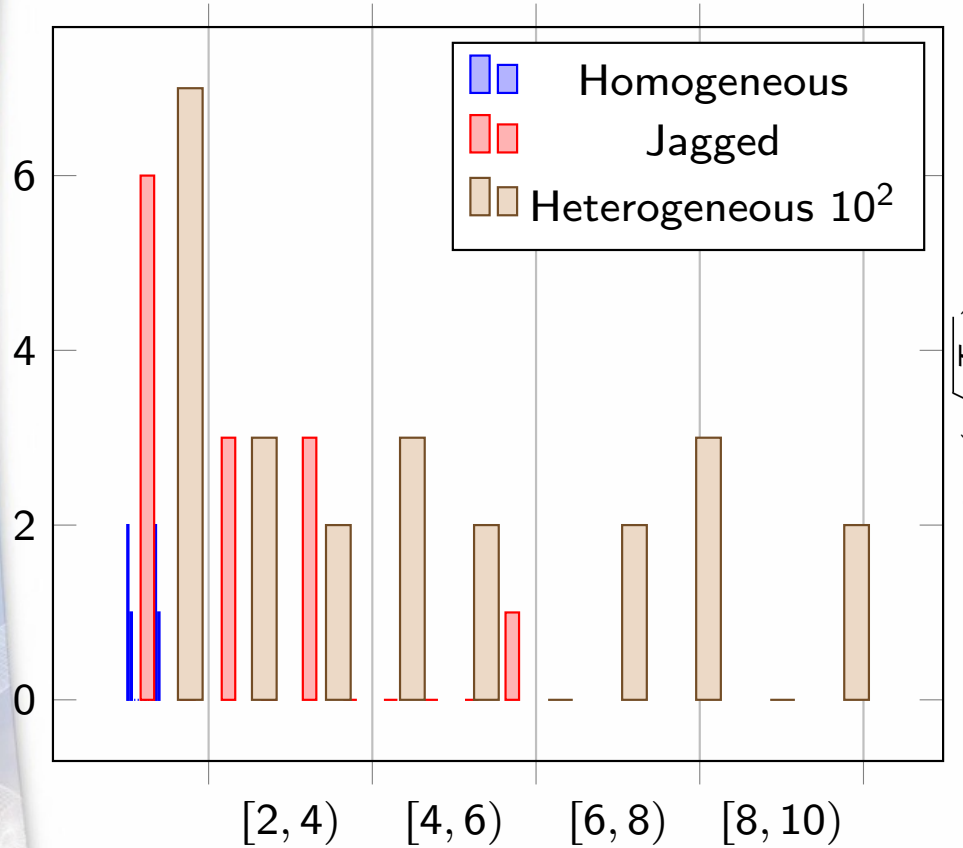


Figure: Spectrum

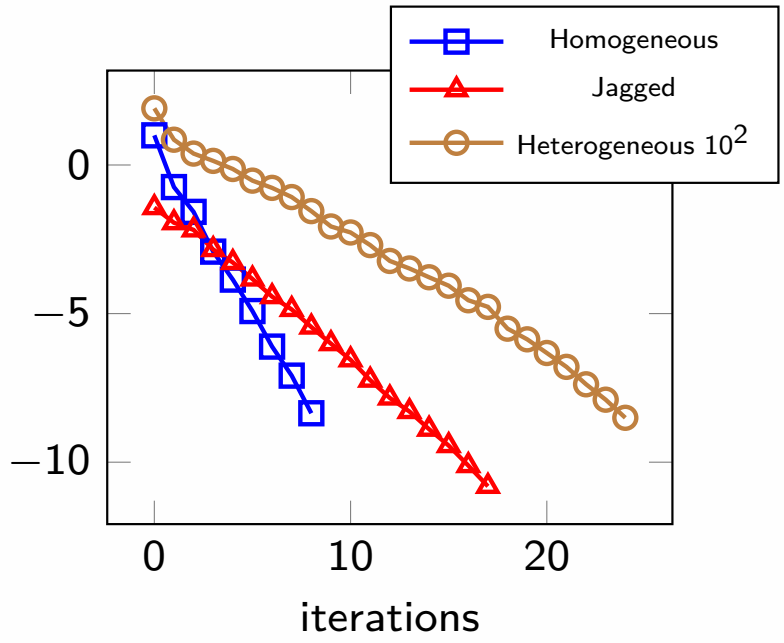


Figure: Convergence

Saint-Venant principle is not enough ...

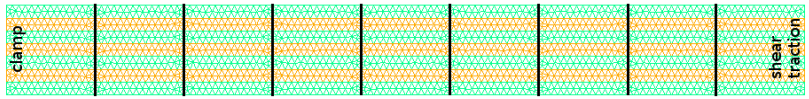


Figure: Beam with straight decomposition

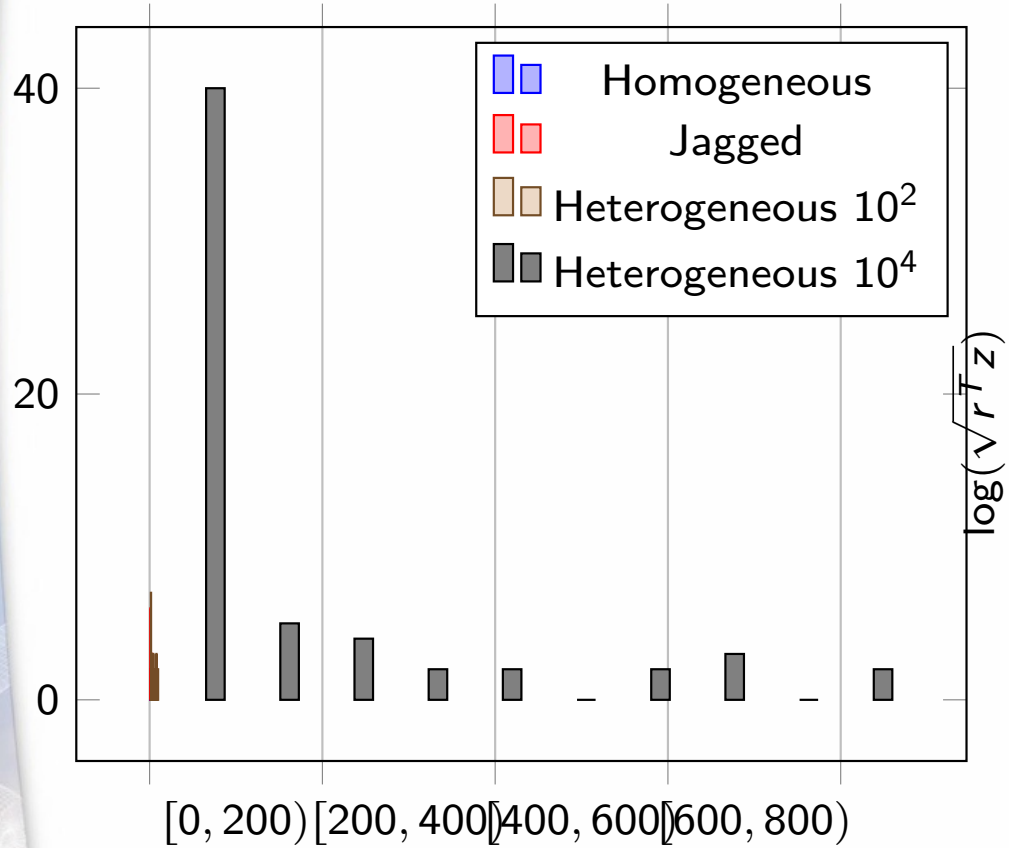


Figure: Spectrum

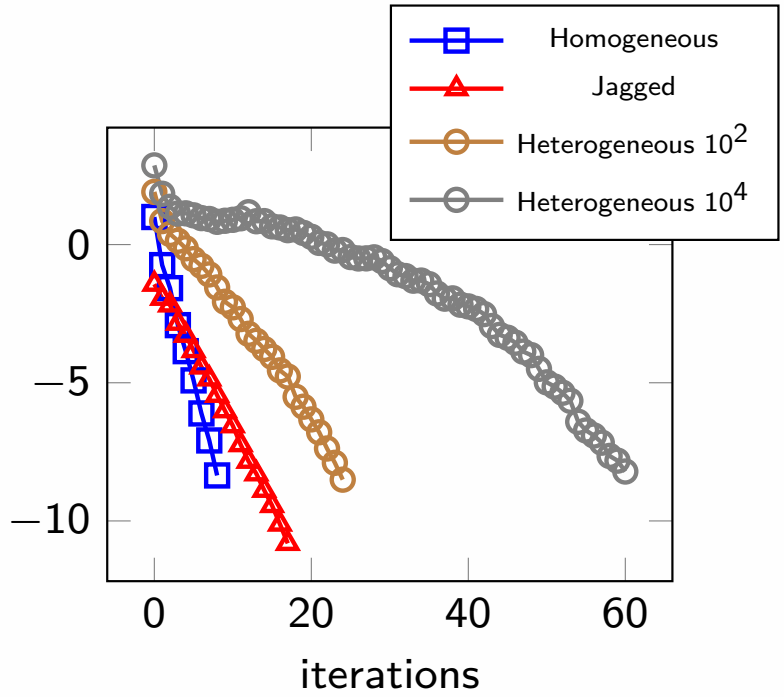
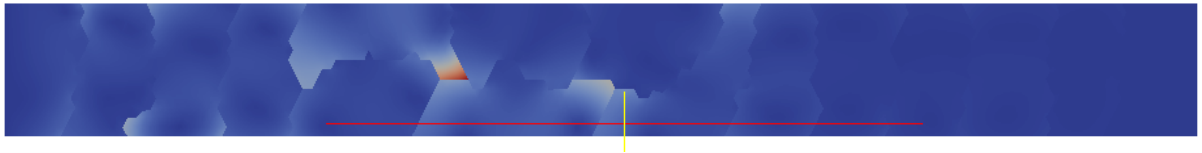


Figure: Convergence

. . . when local features have long-range effects

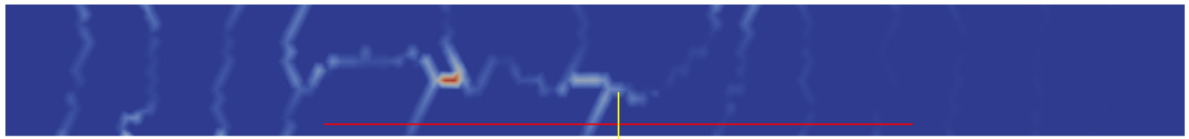
but cures exist

Decomposition with irregular domains

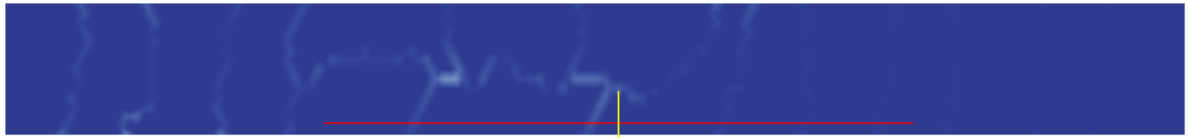


Local contributions to the search direction
The irregularity triggers unnecessary local effects.

One solution is to predict bad modes and removes them a priori (Geneo)
another is to find the optimal combination at each iteration (FETI-S)



Classical Z



Optimized Z

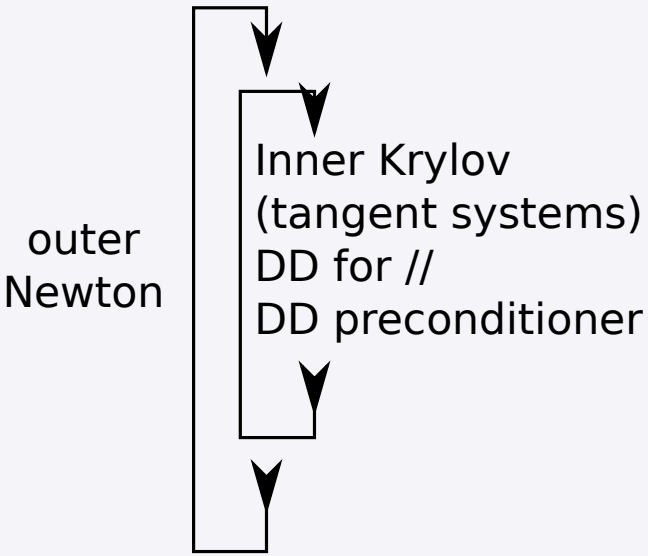
Extension to nonlinear problems

Newton-Krylov-Schur approach

Find **global** displacement u such that

$$f_{int}(u) + f_{ext} = 0$$

linearize and solve with DD.

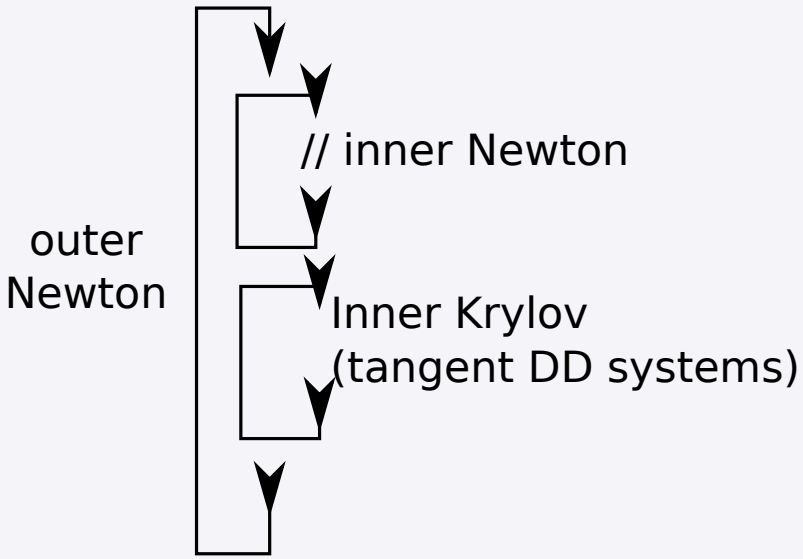


Schur-Newton-Krylov approach

Find **interface** displacement $\mathbf{u_b}$ such that

$$\begin{cases} \forall s,\ f^s_{int}(u^s) + f^s_{ext} = 0 \\ u^s = \mathbf{u_b} \text{ on the interface} \\ \text{nodal reactions are balanced} \end{cases}$$

linearize.



The objective is to replace Krylov iterations (exchanges) by independent inner Newton iterations.

Performance

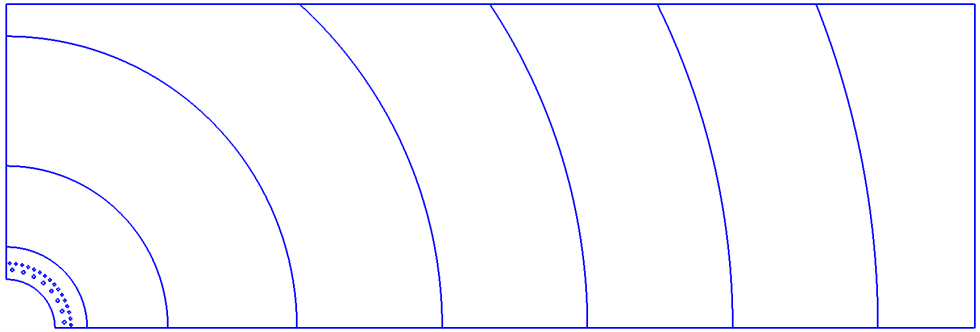


Figure: Localized plasticity problem

Increment of loading	0.2	0.4	0.8	1	1.3
Spread of nonlinearity	Elastic	1SD plastifies		Several SD plastify	
Primal/Classic	1	1	0.75	0.8125	0.815
Dual/Classic	0.5	0.75	0.833	1.25	2.96
Mixed/Classic, $\mathbf{Q} = \mathbf{K}_{bb}$	0.5	0.75	0.667	0.75	0.778
Mixed/Classic, $\mathbf{Q}$ opti	0.5	0.75	0.583	0.563	0.667

Table: Ratios of cumulated global iterations between nonlinearly localized and classic methods

Performance

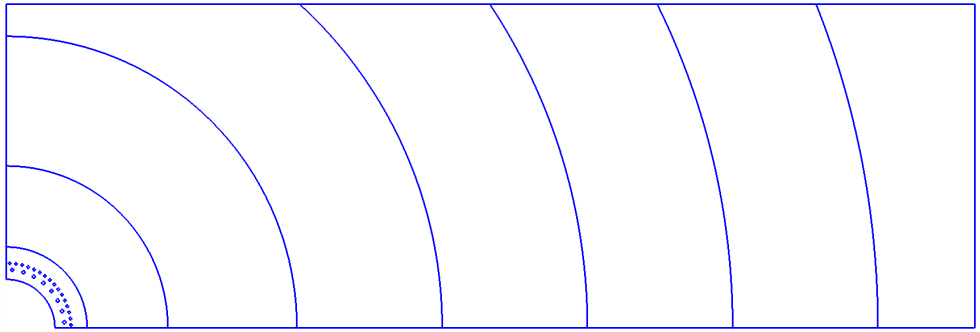


Figure: Localized plasticity problem

Increment of loading	0.2	0.4	0.8	1	1.3
Spread of nonlinearity	Elastic	1SD plastifies		Several SD plastify	
Primal/Classic	1	1	0.745	0.806	0.809
Dual/Classic	0.474	0.701	0.766	1.159	2.734
Mixed/Classic, $\mathbf{Q} = \mathbf{K}_{bb}$	0.5	0.753	0.664	0.745	0.775
Mixed/Classic, $\mathbf{Q}$ opti	0.5	0.753	0.579	0.557	0.662

Table: Ratios of cumulated Krylov iterations between nonlinearly localized and classic methods

Performance

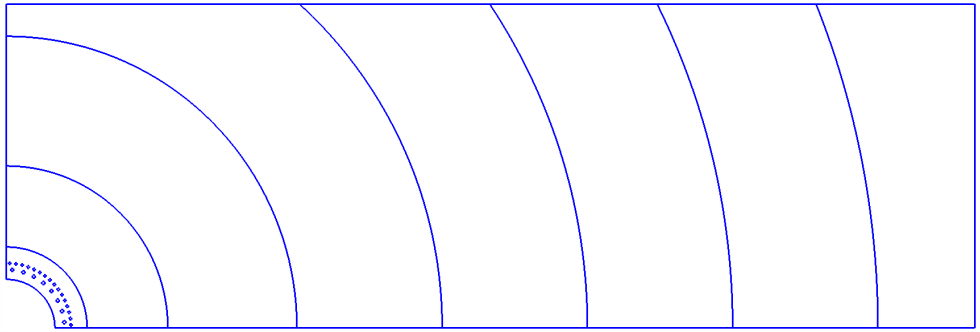


Figure: Localized plasticity problem

Increment of loading	0.2	0.4	0.8	1	1.3
Spread of nonlinearity	Elastic	1SD plastifies		Several SD plastify	
Primal/Classic	1.5	1.75	2	2.438	2.778
Dual/Classic	0.5	1.5	2.667	4.75	14.444
Mixed/Classic, $\mathbf{Q} = \mathbf{K}_{bb}$	0.5	1.25	1.917	2.316	2.667
Mixed/Classic, $\mathbf{Q}$ opti	0	1.25	1.5	1.563	2.222

Table: Ratios of cumulated local iterations between nonlinearly localized and classic methods

Risks

Depending of the chosen boundary condition and of the nature of the nonlinearity their may be limits on the domains of existence of local Schur complements (Dirichlet-to-Neumann or Neumann-to-Dirichlet or Robin-to-Dirichlet).
In that case local Newton iterations would be waste of time.

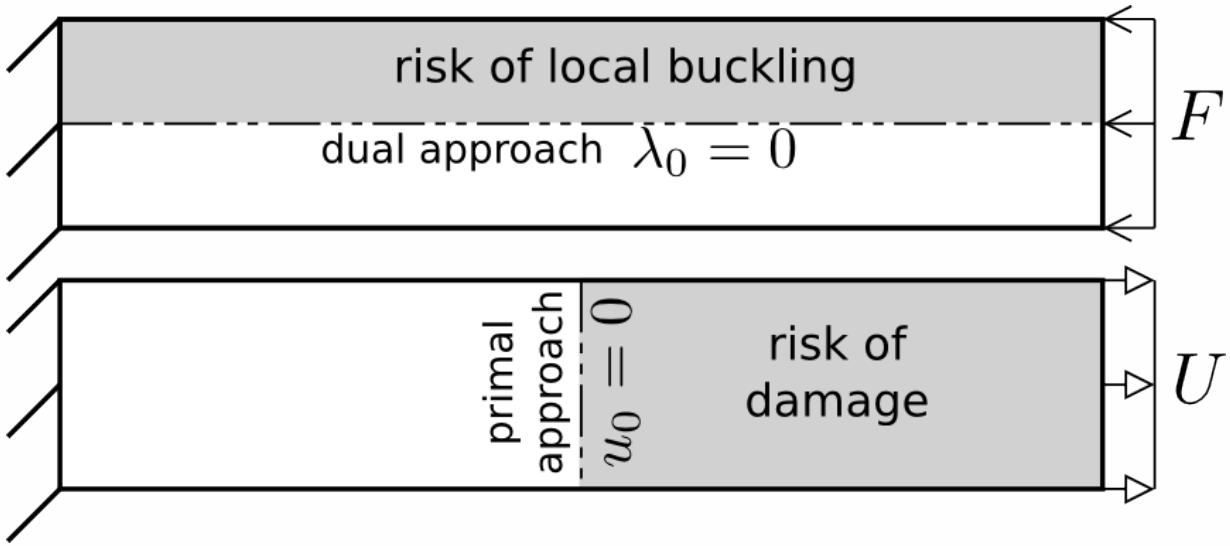


Figure: Two critical cases

The expected cure is finding good mixed (Robin) conditions.

Verification with DD

Objective

- Verification aims at insuring that the underlying (FE) discretization is sufficient to obtain a certain quality in the computation.
- The quality is measured through a global error norm or through quantities of interest.
- Verification is expensive, and domain decomposition methods enable to cut the cost.
- Basically we need to reconstruct fields in H^1 and H_{div}

Separation of the contribution to the error

$$\frac{\sum_s R_N^s(w)}{|||w|||} - |r| \leqslant |||u - u_D||| \leqslant \sqrt{\sum_s \mathrm{e}_{\mathrm{cr}}^2(u_N^s, \hat{\sigma}_N^s)} + |r|$$

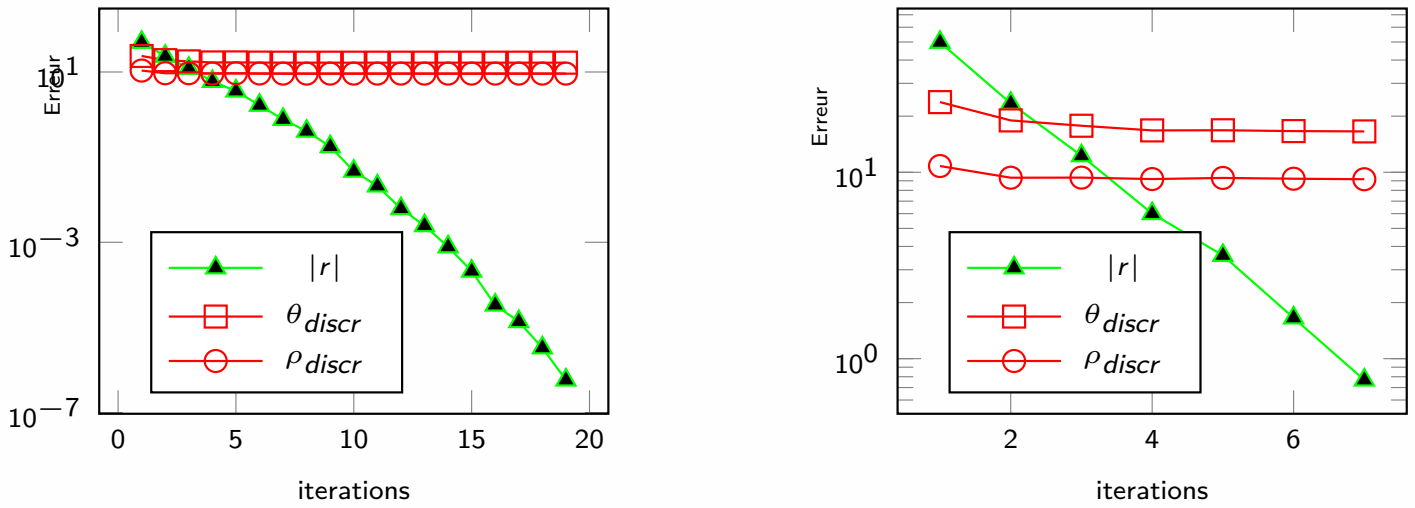


Figure: Enveloppe de l'erreur due à la discrétisation et évolution du résidu

Modeling with domain decomposition methods

The non-overlapping way

When integrated early in the design phase, domain decomposition can be used to model the system as subdomains connected by functional interfaces.
This way we can model:

- Assemblies with contact, friction, adhesion
- Cohesive interfaces like in laminated composites
- Perfect joints (of course)

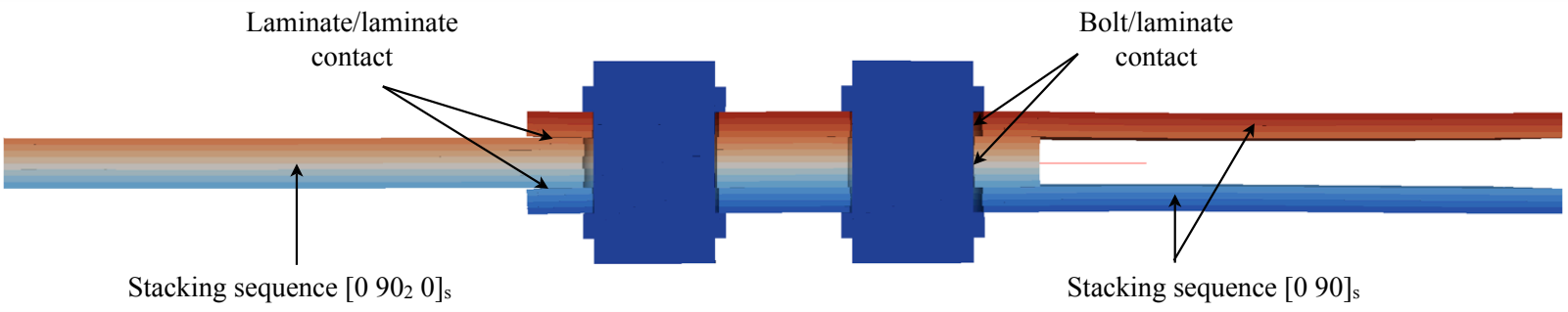


Figure: Bolted composite plates (16 plies).

Modeling with domain decomposition methods

The non-overlapping way

When integrated early in the design phase, domain decomposition can be used to model the system as subdomains connected by functional interfaces.
This way we can model:

- Assemblies with contact, friction, adhesion
- Cohesive interfaces like in laminated composites
- Perfect joints (of course)

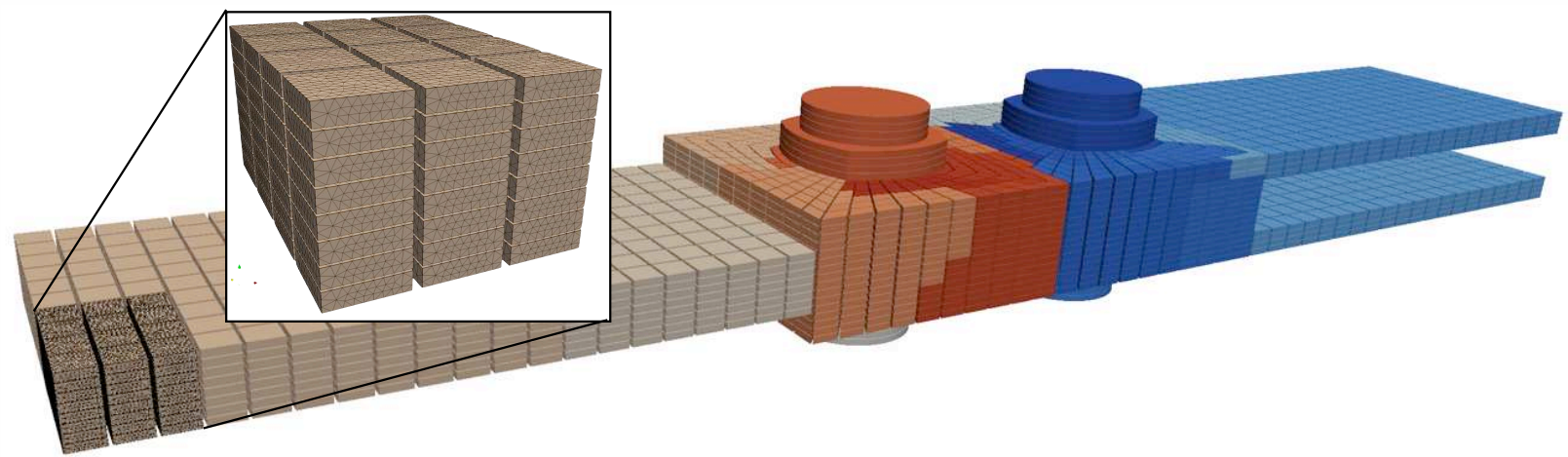


Figure: Discretization ($12\,10^6$ dofs), 10 600 subdomains, 29 CPUs

When integrated early in the design phase, domain decomposition can be used to model the system as subdomains connected by functional interfaces.
This way we can model:

- Assemblies with contact, friction, adhesion
- Cohesive interfaces like in laminated composites
- Perfect joints (of course)

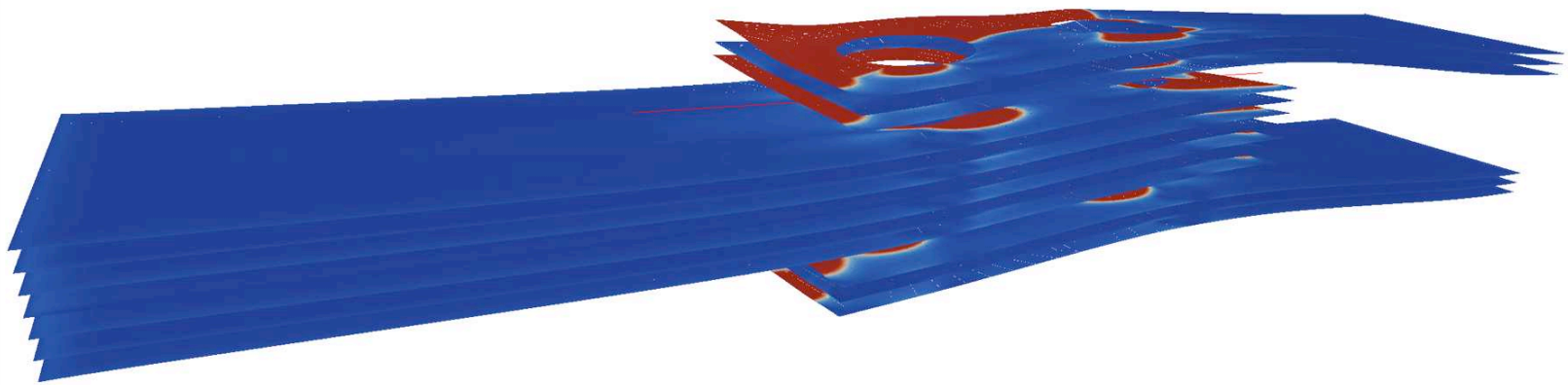


Figure: Damage in the interfaces after the 70^{th} increment.

Modeling with domain decomposition methods

The overlapping way

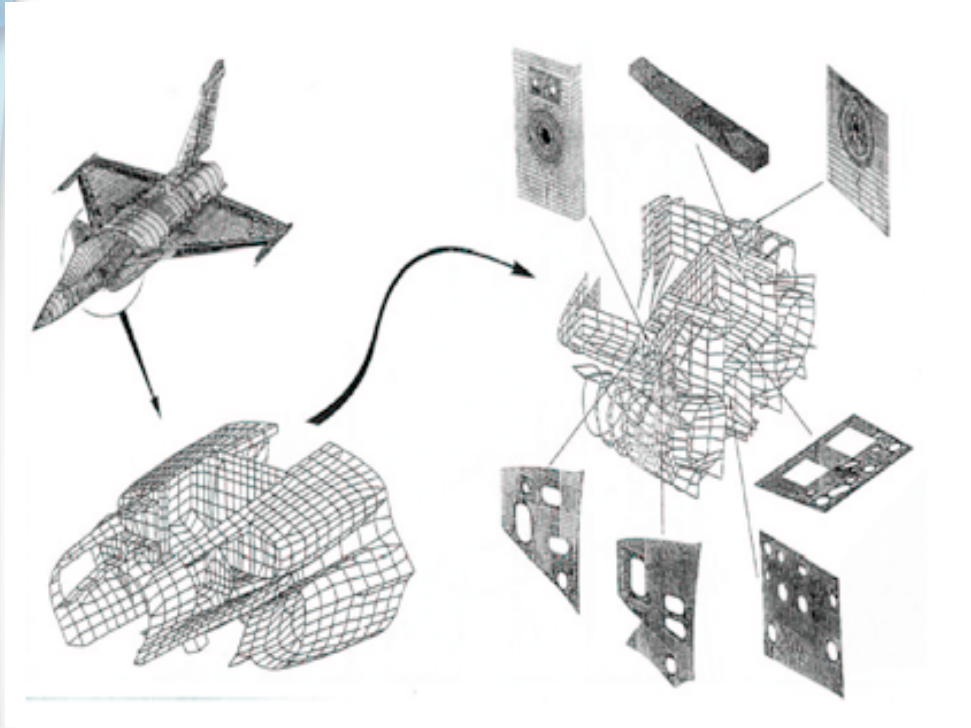
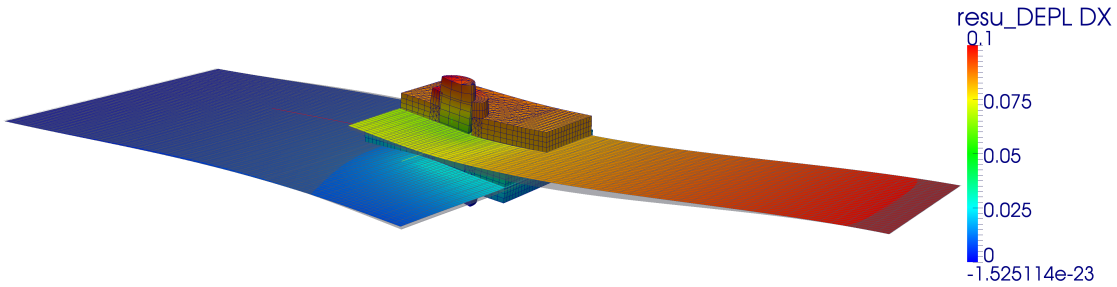


Figure: Structure defined by nested models(Dassault aviation)

Nested models are common in the industry. In order to take into account all interactions (global $\leftrightarrow$ local) DD algorithm can be applied (in nonlinear). The matter in the zones where finer models exist is a parameter equivalent to a Robin bc.



A global model (plate + 1D connector) is patched by a 3D representation of the bolted zone.

Conclusion

- In Latin "I decompose" is *solvo* from which derives "I solve". Indeed solving an equation is understanding how it behaves when decomposed.
- Currently solvers manage billions of degrees of freedom with mild nonlinearity on hundreds of thousand cores, in few minutes.
- In practice, bottlenecks for industrialists are the pre/post-processing.
- Current challenges:
  - Optimal finding and handling of long-range effects / multilevel DD.
  - Robust nonlinear solvers / best bcs.
  - Intelligent splitting, load balancing.
  - Parallel verification and mesh adaption.
- Daniel Rixen's contributions in the field are outstanding:
  - Taking into account heterogeneities in the preconditioner
  - Anticipating long-range effects beyond Saint-Venant's principle
  - Improving solvers to avoid the spread of perturbations
  - Early contribution in today's most efficient family of solvers (FETI-DP)
  - Handling of MPCs which is crucial for industrial applications
  - more to come

Thank you for your attention

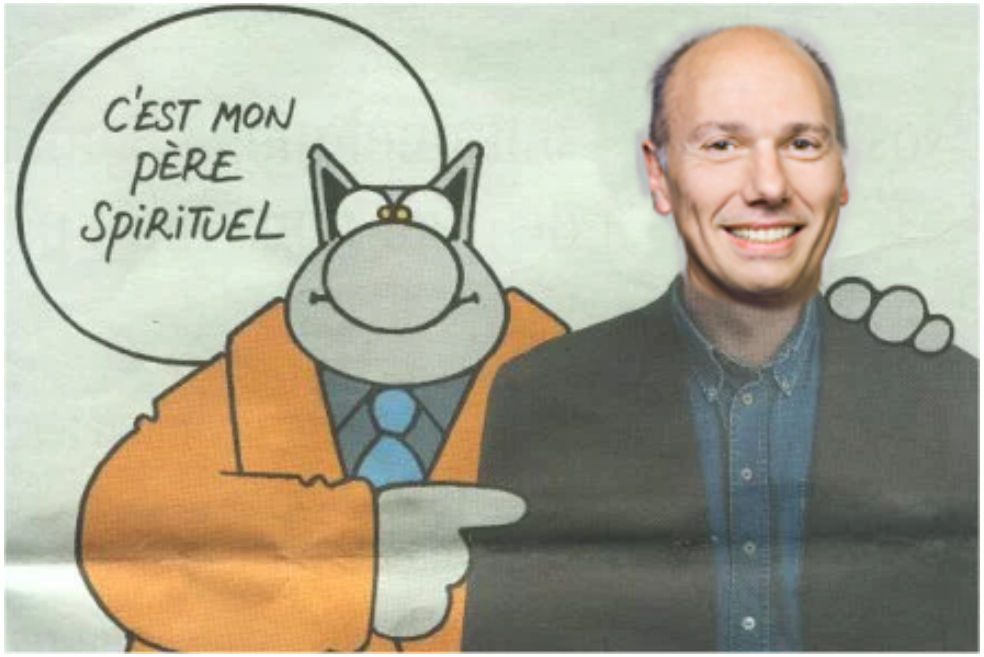


Figure: Made in Belgium (adapted from Gelluck)