RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

ONERA
THE FRENCH AEROSPACE LAB

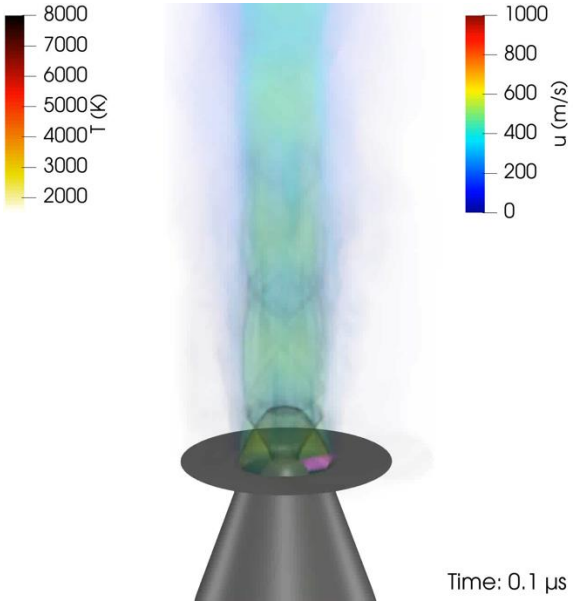
High-order adaptive multistep coupling

L. François[1], A. Simon[1,2], E. Quémerais[1], M. Massot[2], A. Asad[2]

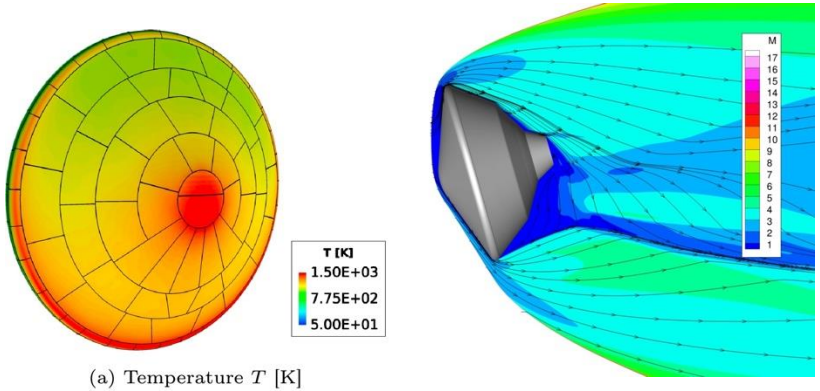
[1] ONERA, DMPE
[2] CMAP, Ecole Polytechnique

Ce document est la propriété de l'ONERA. Il ne peut être communiqué à des tiers et/ou reproduit sans l'autorisation préalable écrite de l'ONERA, et son contenu ne peut être divulgué.
This document and the information contained herein is proprietary information of ONERA and shall not be disclosed or reproduced without the prior authorization of ONERA.

Multiphysics

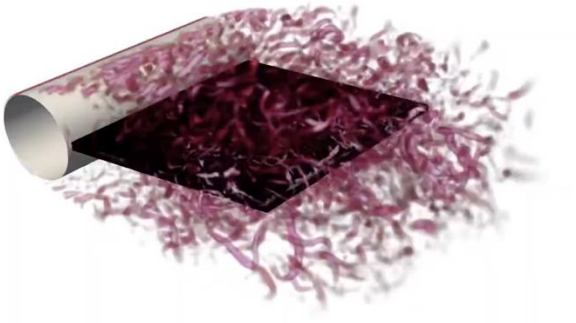


Electric arc in a supersonic injector
A. Bourlet et al (ONERA)



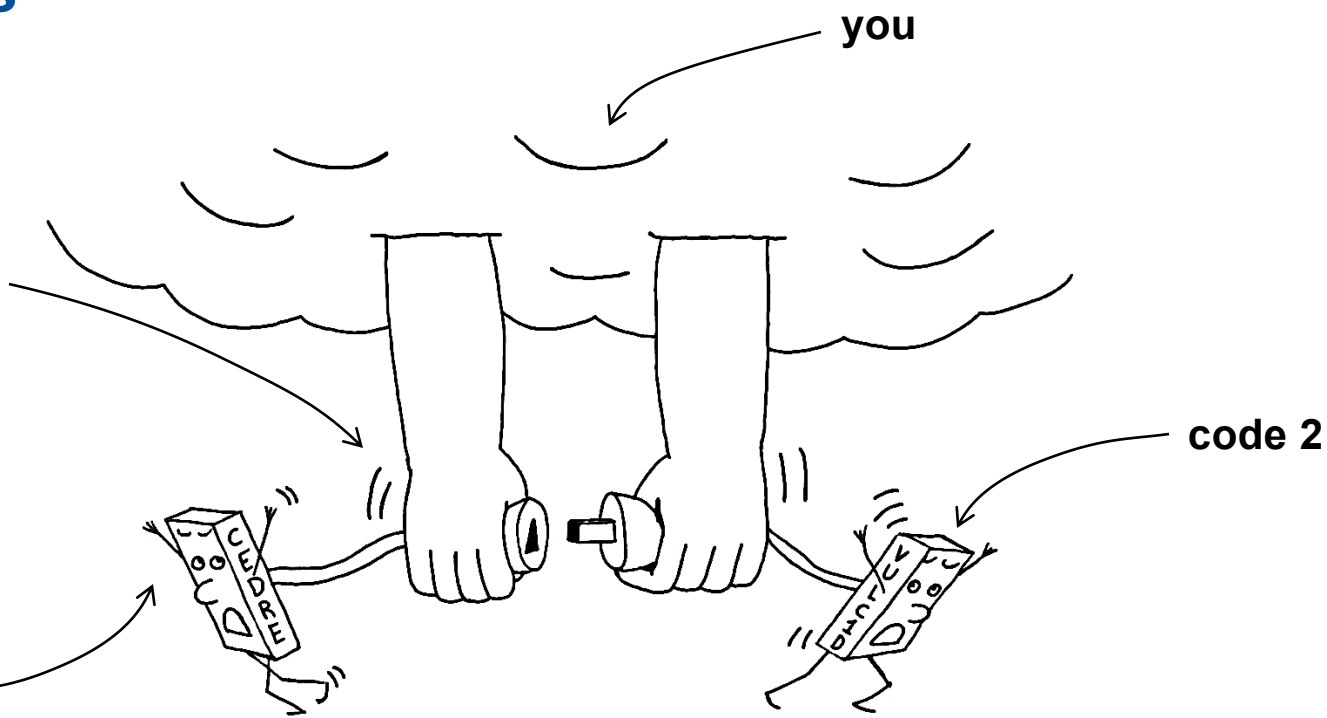
(a) Temperature T [K]

Multidimensional material response simulations of a full-scale tiled ablative heatshield
J. Meurisse et al, NASA



Fluid-structure interaction with Yales2
T. Fabbri et al (CORIA)

Multiphysics



Non-exhaustive list:
- *formulation*
- *code interface*
- *interpolation*
- *coupling stability*
- *conservativity*
- *accuracy*
- *…*

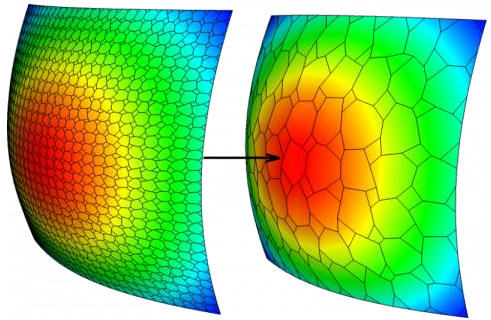
Artist's view of the coupling process

CWIPI – I was made for coupling you !

- **Open-source** coupling library[1]
- **Data transfer** and **spatial interpolation** from mesh to mesh (**volume or surface** coupling)
- **HPC**-oriented (MPI with CPUs, GPU port ongoing[2])
- Any number of codes

- **Multiple interpolation modes**:
  point cloud, conservative mesh-based interpolation,
  high-order curved meshes, custom callbacks…

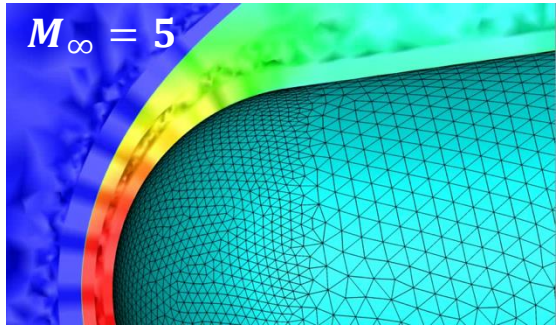
- No pilot task, CWIPI is **embedded** in each code
- C/C++, Fortran, and Python **interfaces**
- Multiple academia and industrial users

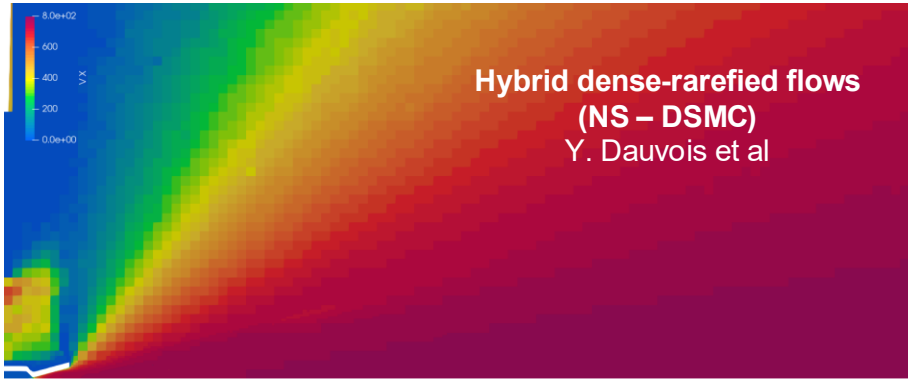


[1] E. Quémerais et al, *github.com/onera/cwipi*
[2] PhD of R. Cazalbou (ONERA/CERFACS) 2023 - 2025

CWIPI - Examples of applications at ONERA

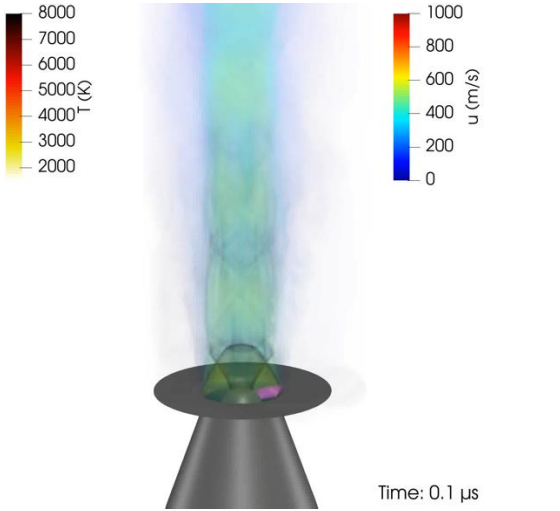
Ablation with moving meshes
N. Perron, N. Dellinger

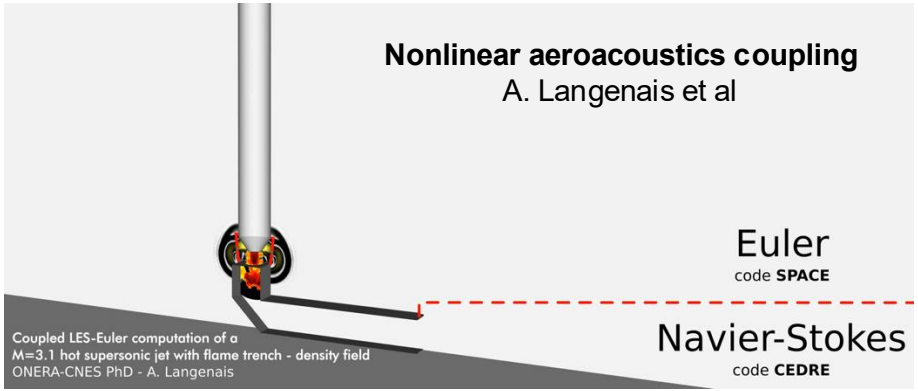




Hybrid dense-rarefied flows (NS – DSMC)
Y. Dauvois et al

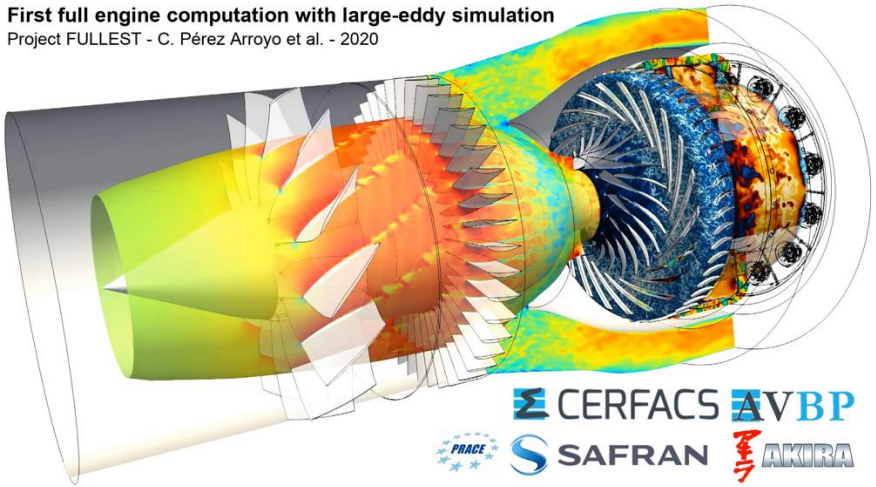
Electric arc in a supersonic flow
A. Bourlet, F. Tholin, A. Vincent, J. Labaune





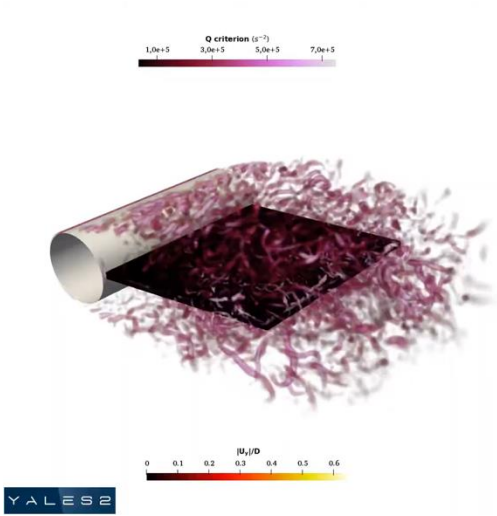
Nonlinear aeroacoustics coupling
A. Langenais et al

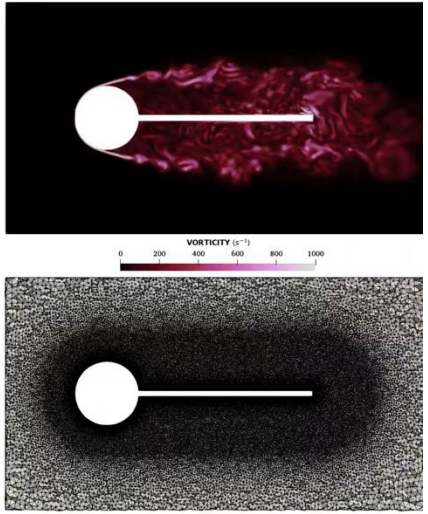
CWIPI - Example of applications outside ONERA



C. Pérez Arroyo et al (CERFACS) [JGPPS, 2020]

- **2 billion cells**
- Coupling of **3 AVBP instances**
- **3D volume coupling**





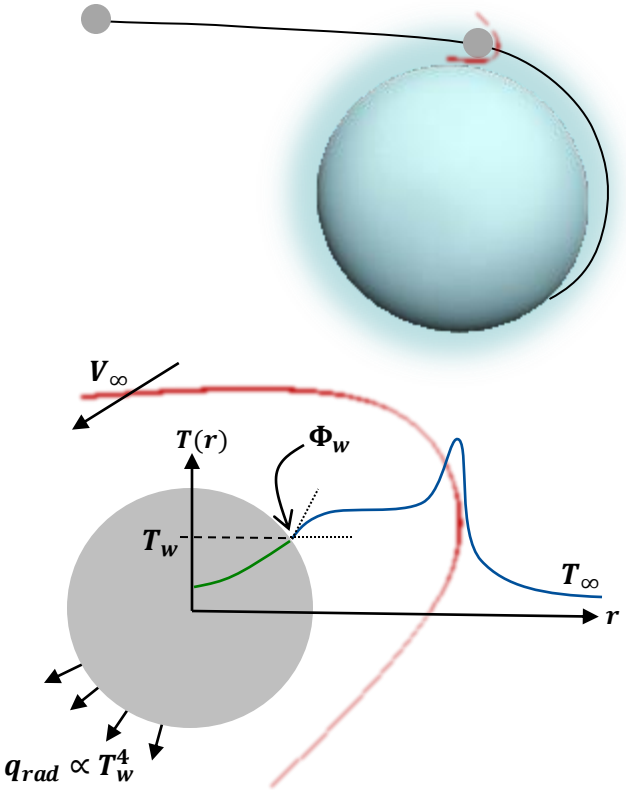
Fluid-structure interaction, T. Fabbri 2022

- **Yales2** CFD solver coupled via CWIPI to a structural solver
- **First-order implicit** coupling solved by **relaxation**
- **Deforming** geometry and mesh

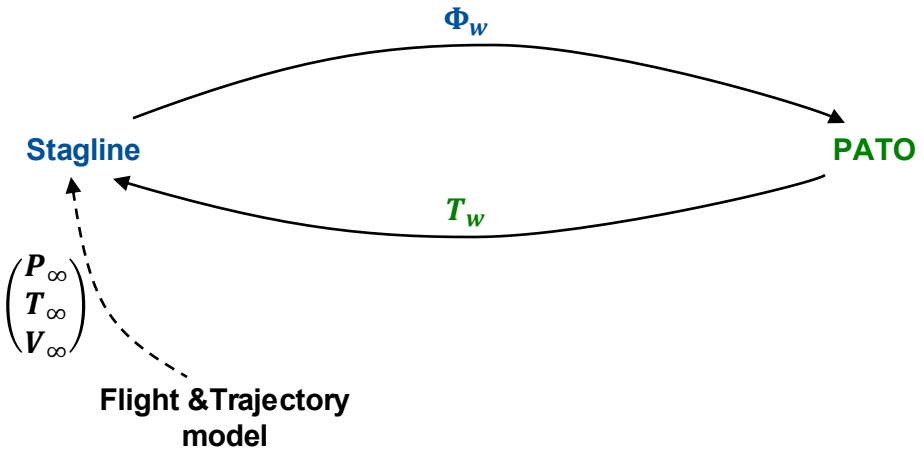
Multiphysics

- Spatial coupling 
- What about the temporal coupling ?

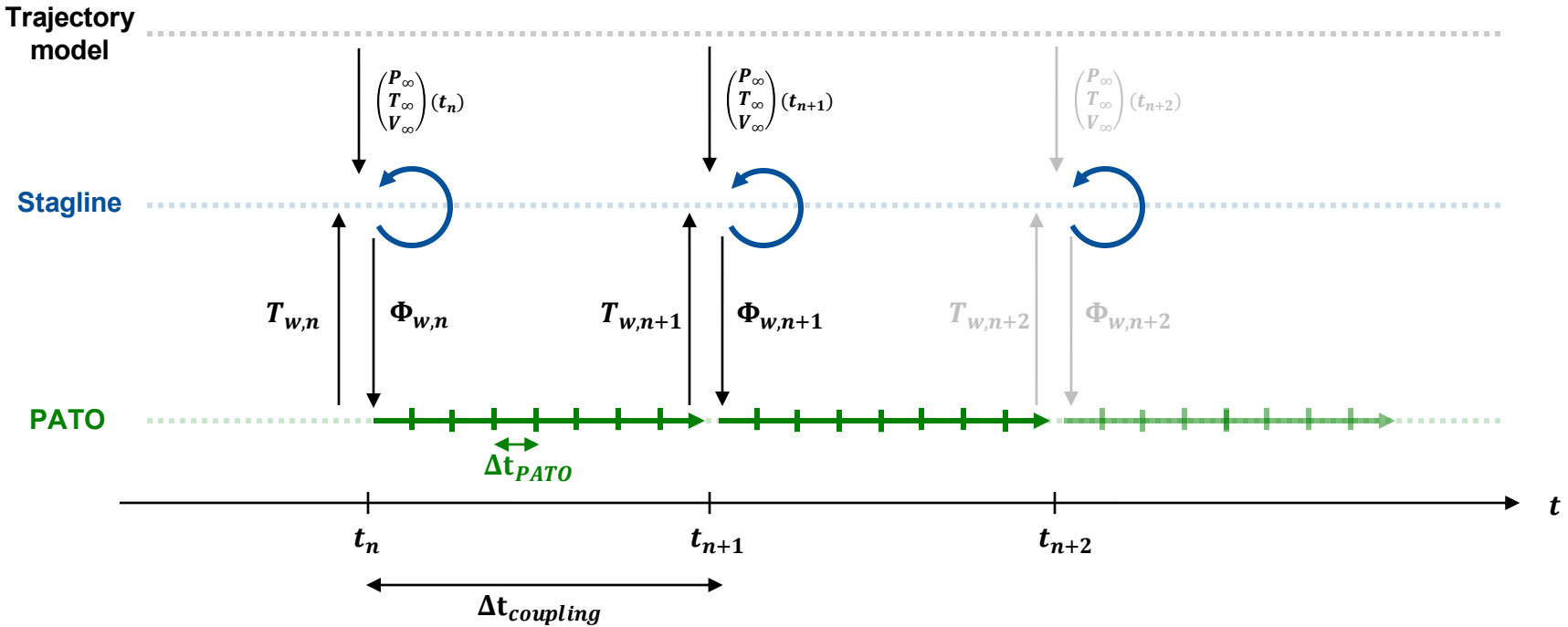
A motivating entry test case



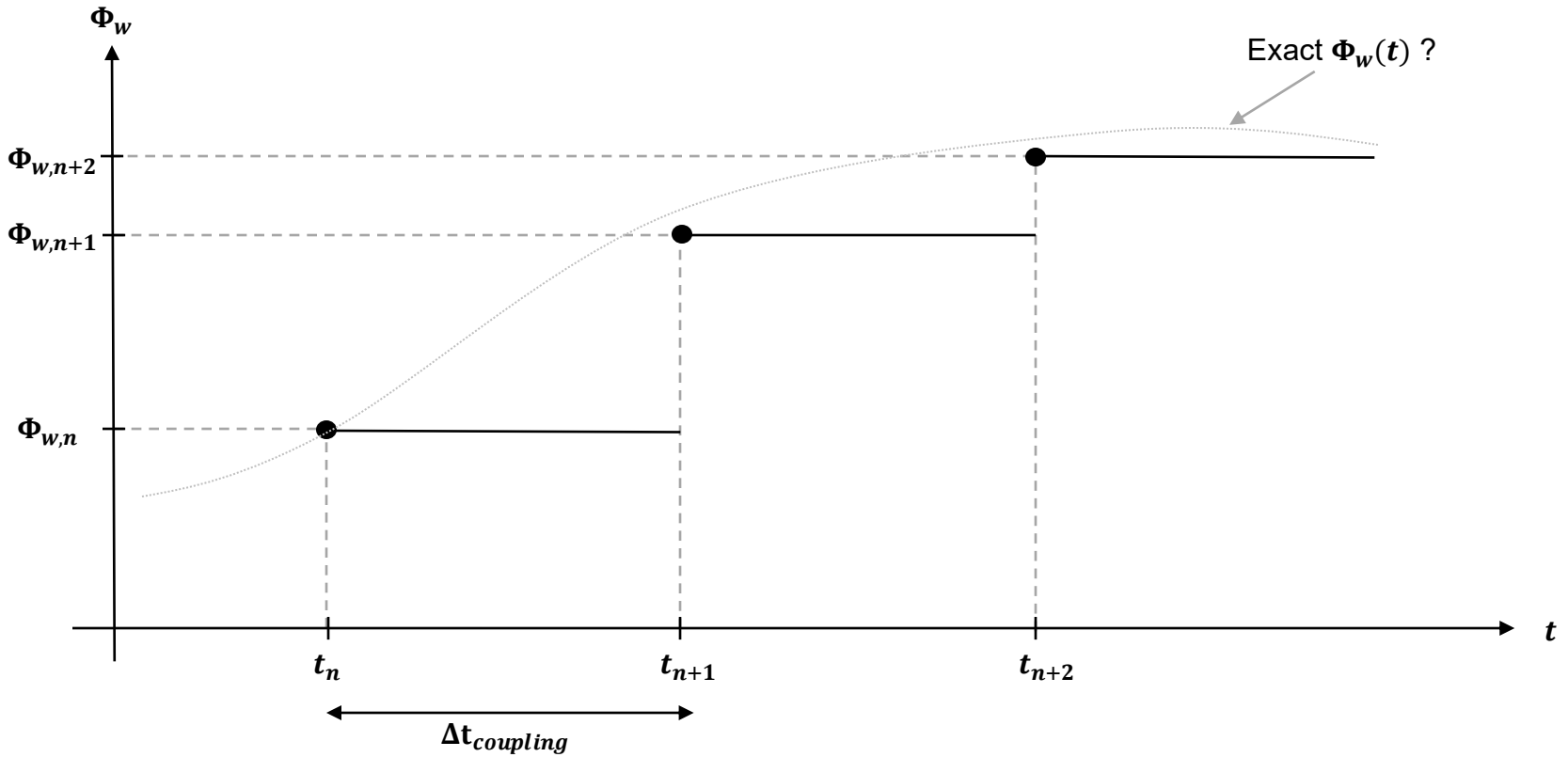
- Compute the transient heating and ablation
- Precomputed flight trajectory
- Coupling of **hypersonic flow** model with a **material model**



Typical time coupling scheme



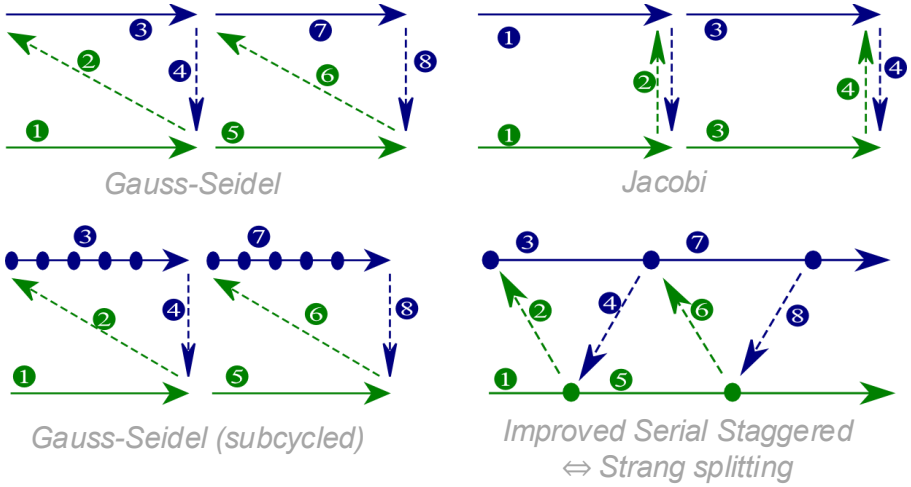
Typical time coupling scheme



Pros and cons

Advantages

- **Simple**
- **Works !**



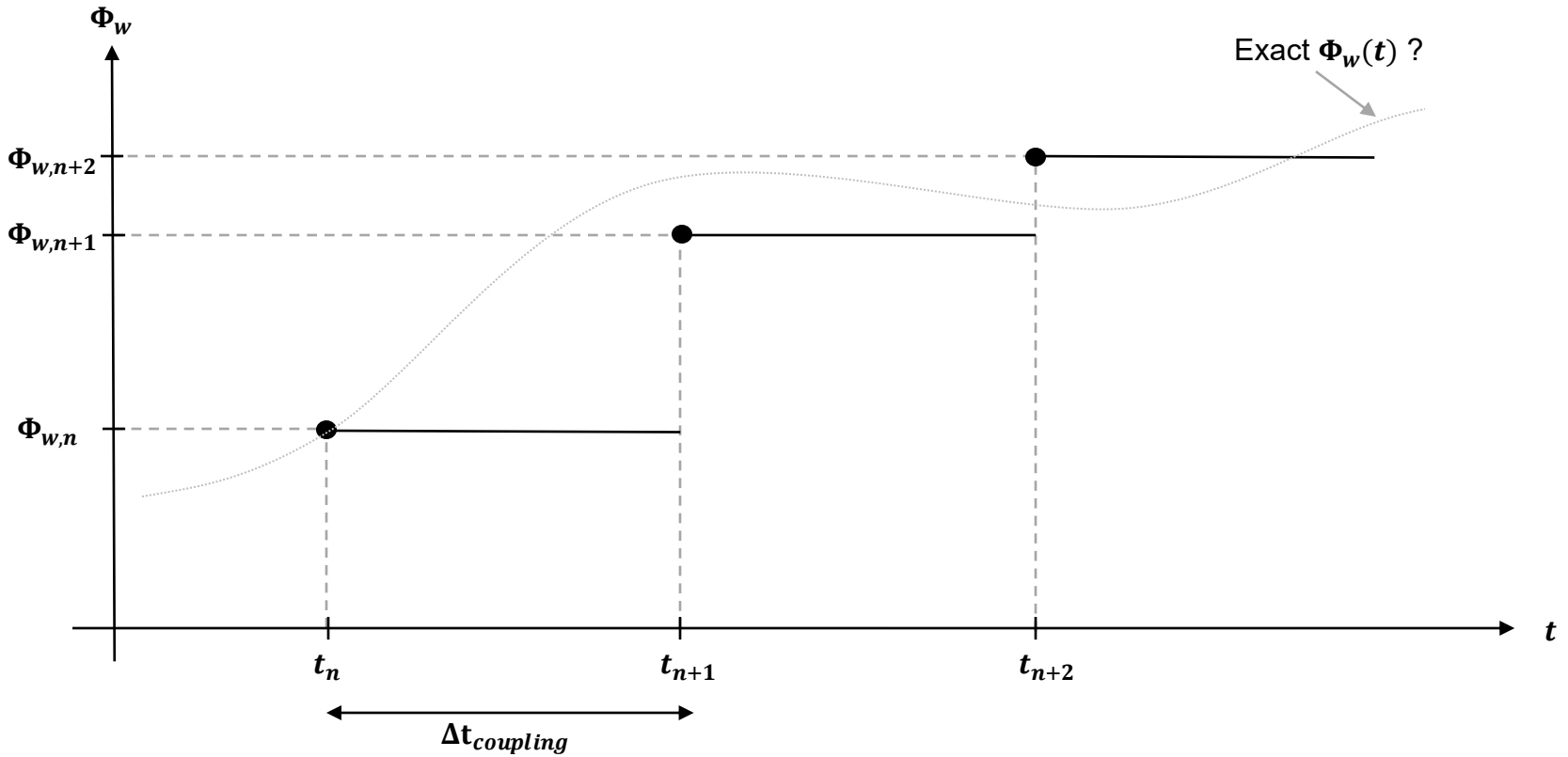
Deficiencies

- **User-defined** for each coupling
- **Explicit**
- **Constant Δt** chosen manually
- **1st-order** accuracy, no **error estimation**
- Implicit **difficult** to implement

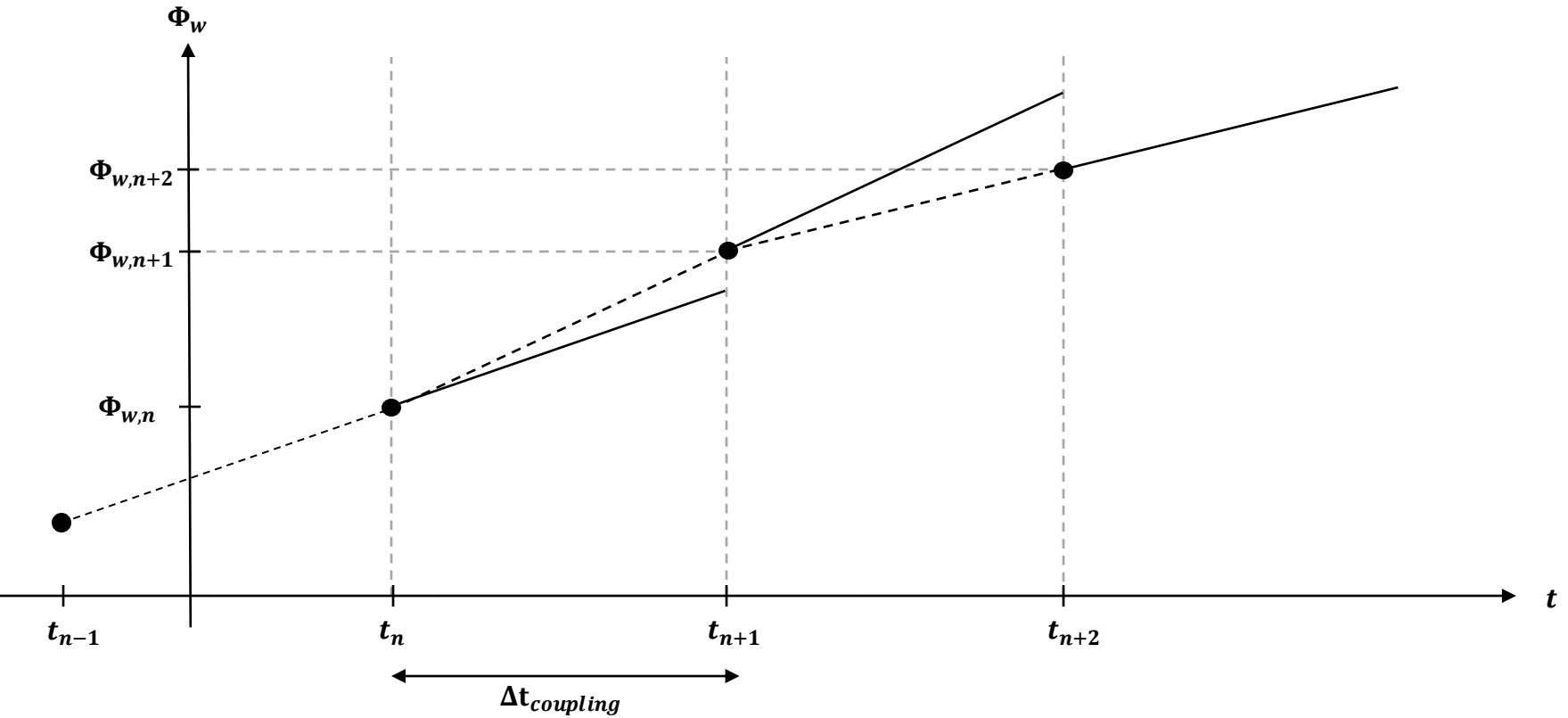
Wishlist ☺

- **Adaptive Δt**
- **High-order accuracy**
- **Streamlined implementation**
- **Easy-to-use implicit variant**

How to improve ?



How to improve ?



How to improve ?

- Replace the 'staircase' approximation of by a **degree-k polynomial** of time, extrapolating $k+1$ past values:

$$\widehat{\boldsymbol{u}}(\boldsymbol{t}) = \mathbf{extrap}(\boldsymbol{u_n}, \boldsymbol{u_{n-1}}, \dots, \boldsymbol{u_{n-k}}, \boldsymbol{t_n}, \dots, \boldsymbol{t_{n-k}})(\boldsymbol{t})$$

- The global coupling error in time is $\mathbf{O}(\boldsymbol{\Delta t^k})$.

Remark: $\boldsymbol{u}$ denotes the vector of coupling variables:

$$\boldsymbol{u_n} = (\boldsymbol{T_{w,n}}, \boldsymbol{\Phi_{w,n}})^t$$

- Time adaptation with the error estimate:

$$\boldsymbol{e_u} = |\widehat{\boldsymbol{u}}(\boldsymbol{t_{n+1}}) - \boldsymbol{u_{n+1}}| = \mathbf{O}(\boldsymbol{\Delta t^{k+1}})$$

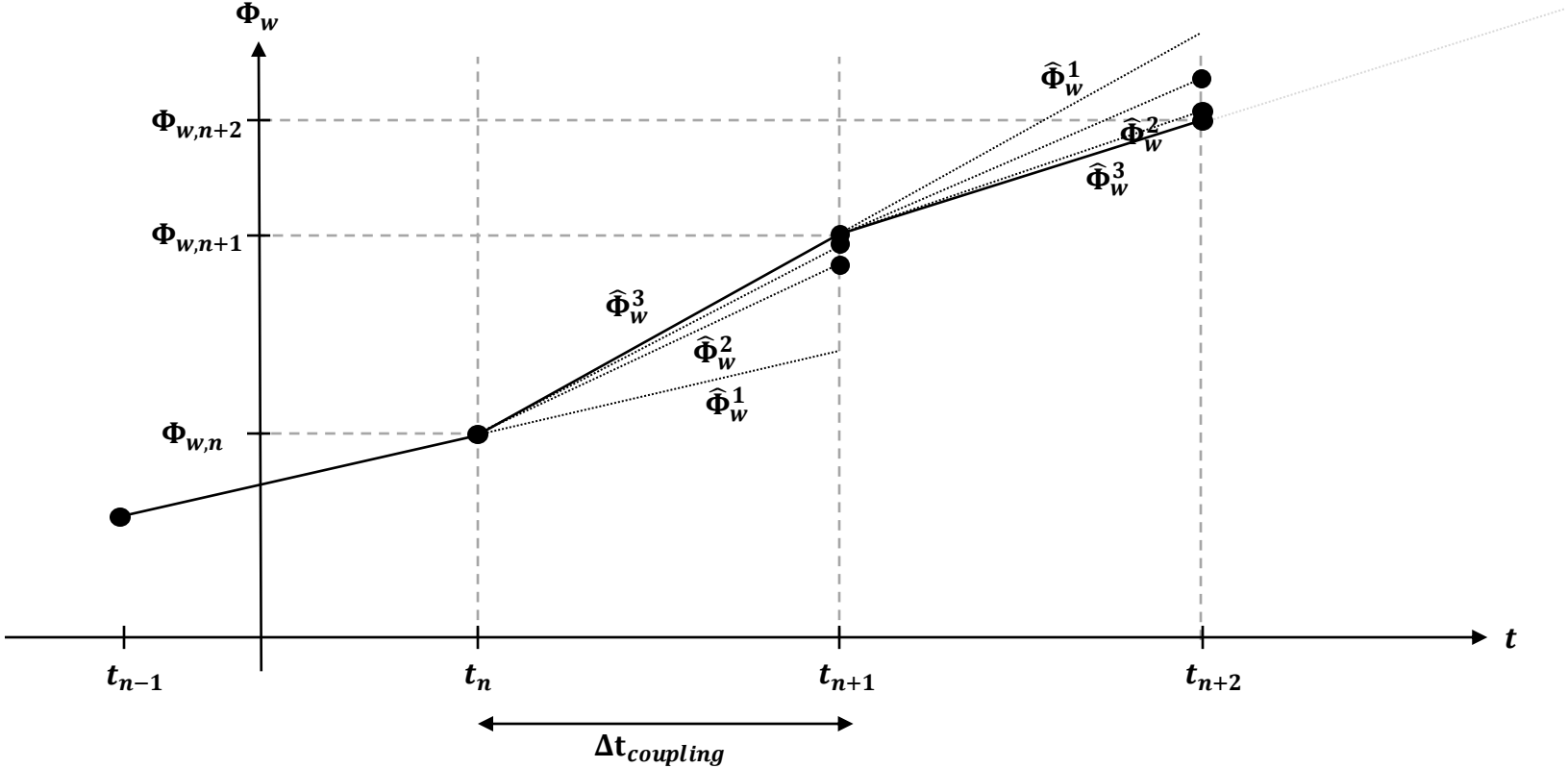
Example: First step with $\boldsymbol{e_u}(\boldsymbol{\Delta t_1}) > \mathbf{tol}$*, the optimal time step is* $\boldsymbol{\Delta t_{opt}} = \boldsymbol{\Delta t_1}\left(\frac{\boldsymbol{tol}}{\boldsymbol{e_u}(\boldsymbol{\Delta t_1})}\right)^{\frac{1}{k+1}}$

- Multistep step nature (e.g. Adams-Bashforth and BDF methods)

→ **"Multistep coupling scheme"[1,2]**

[1] *L. François, M. Massot, "Multistep interface coupling for high-order adaptive black-box multiphysics simulations", Coupled2023*
[2] *Antoine E. Simon, L. François, M. Massot, "High-order multistep coupling: convergence, stability and PDE application", submitted to CRAS*

Towards an implicit coupling scheme

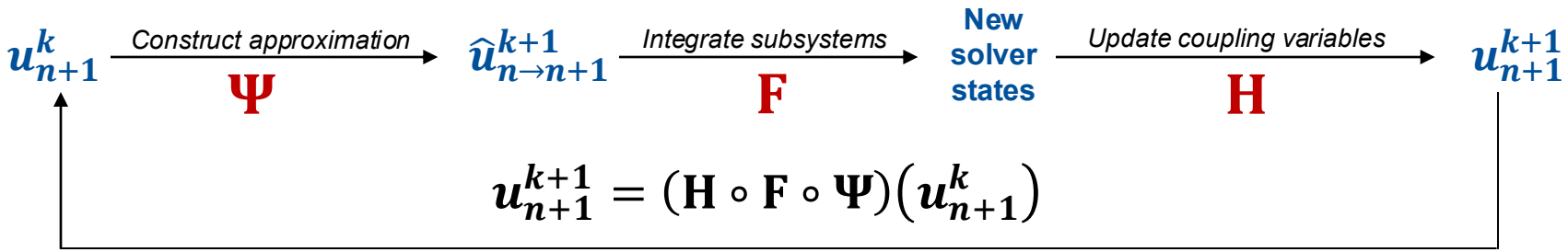


Towards an implicit coupling scheme

- Replace the polynomial approximation becomes an *interpolation* of the previous iteration:

$$\hat{u}(\mathbf{t}) = \mathbf{extrap}(u_{n+1}, u_n, \dots, u_{n-k+1}, \mathbf{t}_{n+1}, \mathbf{t}_n, \dots, \mathbf{t}_{n-k+1})(\mathbf{t})$$

- Fixed-point approach:



- Equivalent root-finding problem:

$$\mathbf{0} = u - (\mathbf{H} \circ \mathbf{F} \circ \Psi)(\mathbf{u})$$

Remark: your codes usually already implement $\mathbf{H} \circ \mathbf{F}$

Theoretical analysis

The multistep coupling scheme is zero-stable and consistent, thus **convergent**.

The local error is $\boldsymbol{O(\Delta t^{k+1})}$.

The global error is $\boldsymbol{O(\Delta t^{k})}$ for k-degree polynomials.

[1] *L. François, "Multiphysics modelling and simulation of the ignition transient in solid rocket motors", PhD dissertation, 2022*
[2] *L. François, M. Massot, "Multistep interface coupling for high-order adaptive black-box multiphysics simulations", Coupled2023*
[3] *Antoine E. Simon, L. François, M. Massot, "High-order multistep coupling: convergence, stability and PDE application", submitted to CRAS*

Stability

- **Linear** setting, with a 2-dimensional Dahlquist equation:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} \lambda_1 & \lambda_{12} \\ \lambda_{21} & \lambda_2 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

- Exact subsystem integration :

$$y_1^{n+1} = y_1^n + \int_{t^n}^{t^{n+1}} \lambda_1 y_1(t) + \lambda_{12}\, \widehat{y_2}(t)\, dt$$

$$y_2^{n+1} = y_2^n + \int_{t^n}^{t^{n+1}} \lambda_2 y_2(t) + \lambda_{21}\, \widehat{y_1}(t)\, dt$$

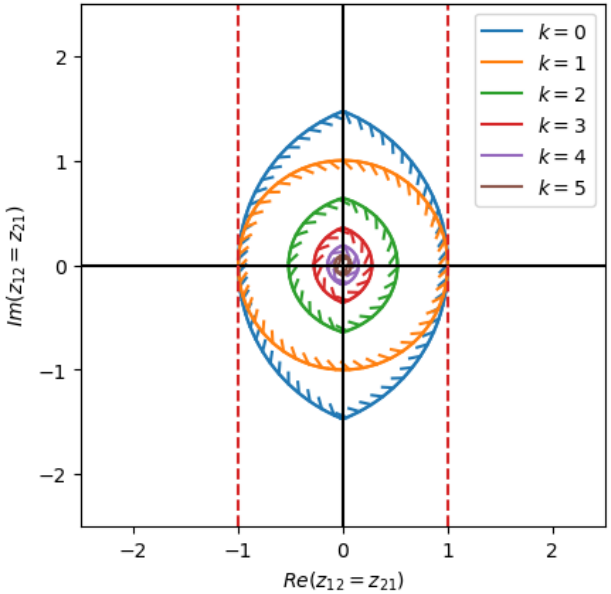
- Introducing $z_i = \Delta t\, \lambda_i$, we obtain a **matrix recurrence** :

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}^{n+1} = M(\mathrm{z}_1, \mathrm{z}_2, \mathrm{z}_{12}, z_{21}) \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}^n$$

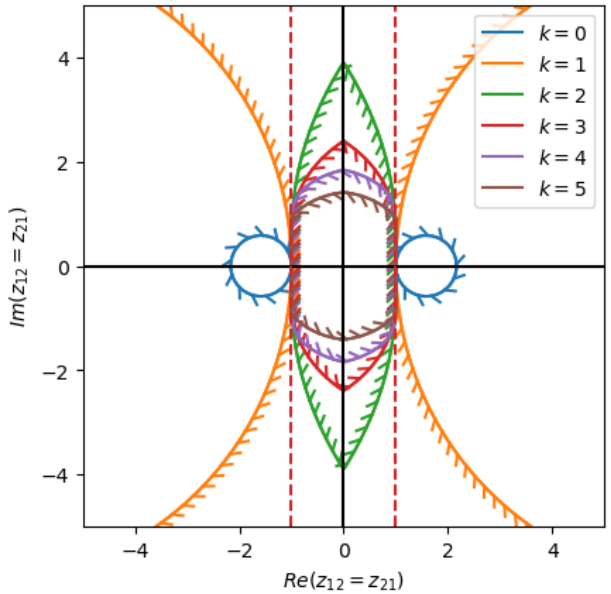
- Stable if spectral radius $\rho(M) \leq 1$

Stability

- With complex values, the stability domain is 8D !
  - → need to make cuts
  - → here, cut at $z_1 = z_2 = -1$ and $z_{12} = z_{21} \in \mathbb{C}$



Explicit



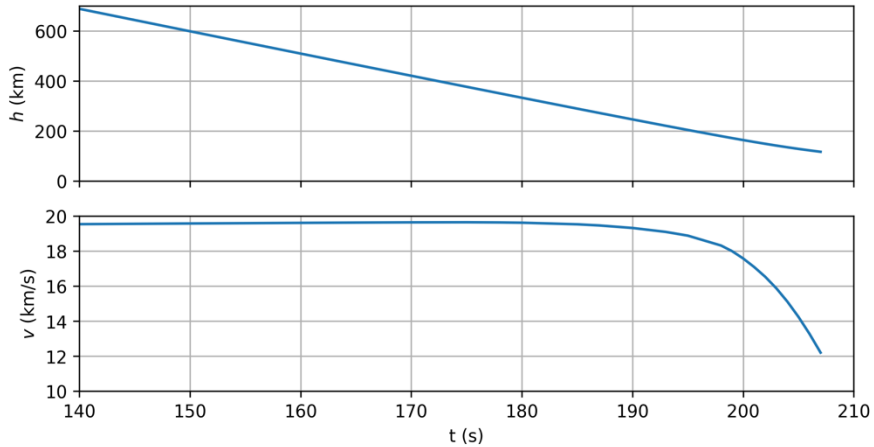
Implicit

Back to the Uranus entry trajectory

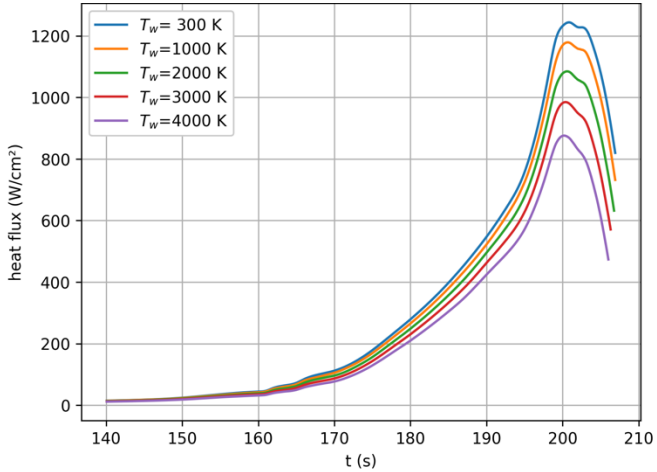
- **Uranus Orbiter and Probe** reentry
- 2D **tabulation** of Stagline to speed up experiments $\Phi_w(t, T_w)$, including fully catalytic effects
- 1D spherical material heat solver with surface radiation, no recession / blowing, $r = 1$ m
- **Rhapsopy** Python demonstrator [1] of the multistep coupling:

  Really **H**igh-order **A**da**P**tive coupling for **S**imulation **O**rchestration in **Py**thon

  github.com/hpc-maths/Rhapsopy



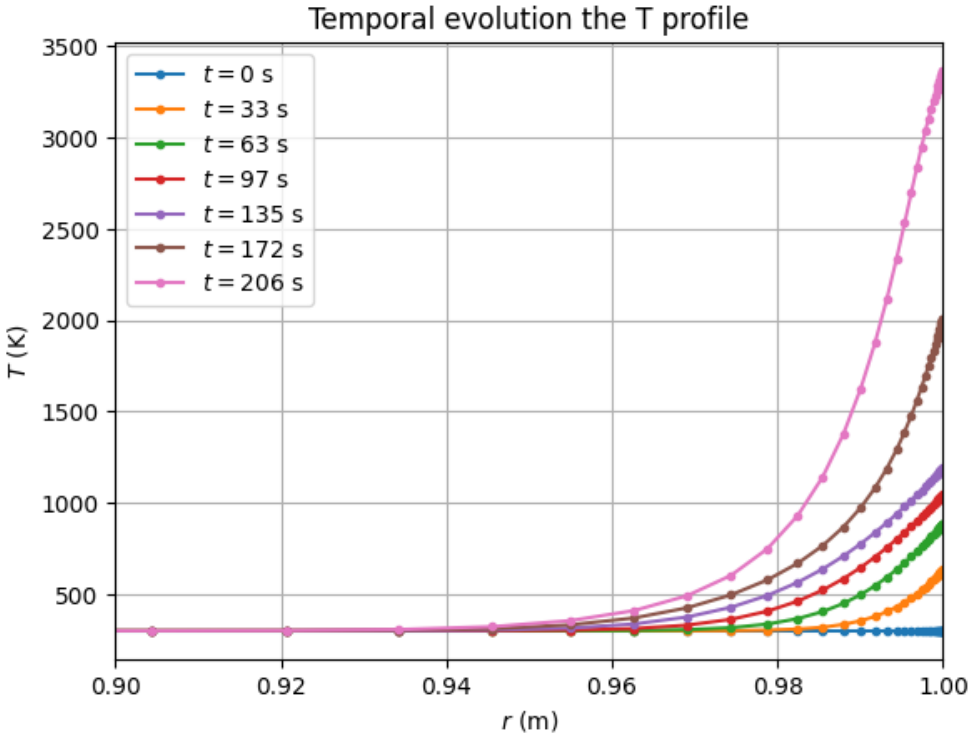
Reentry trajectory



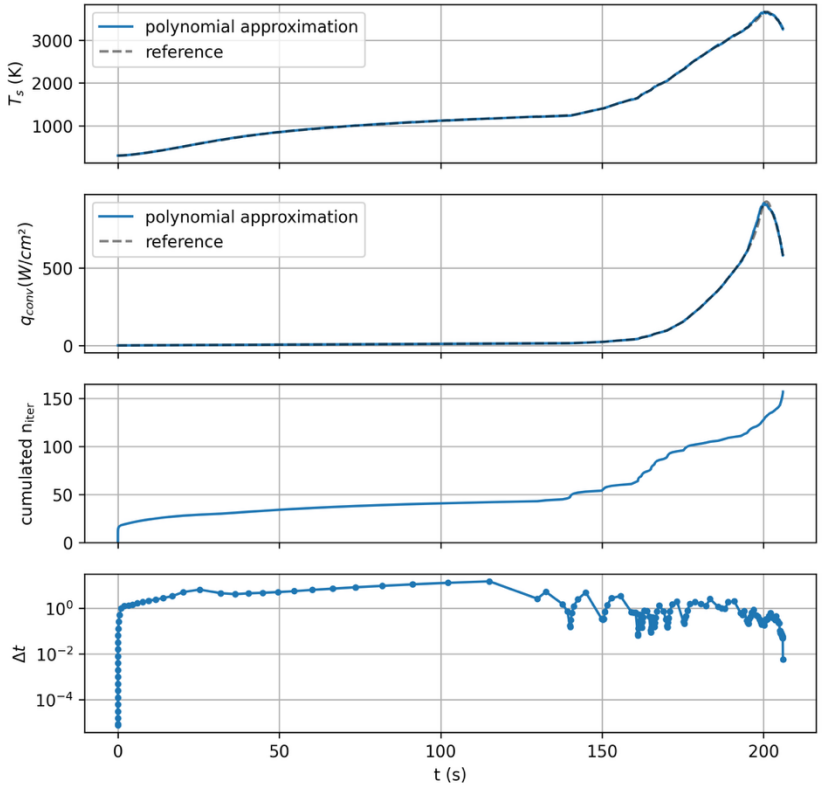
Stagline heat flux tabulation

[1] *L. François, M. Massot, "Multistep interface coupling for high-order adaptive black-box multiphysics simulations", Coupled2023*

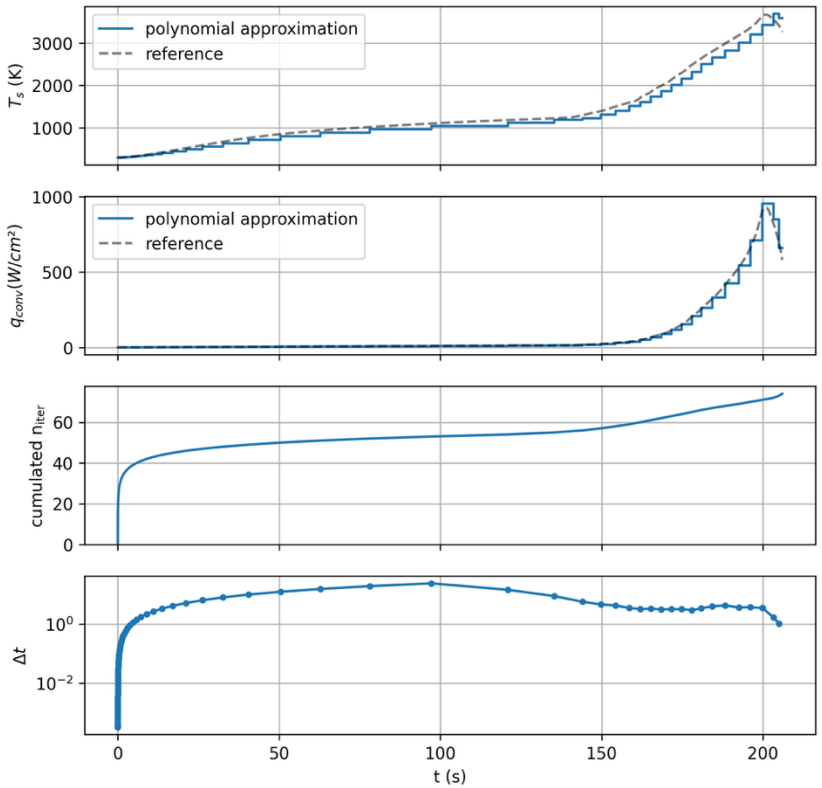
Full entry trajectory



Full entry trajectory

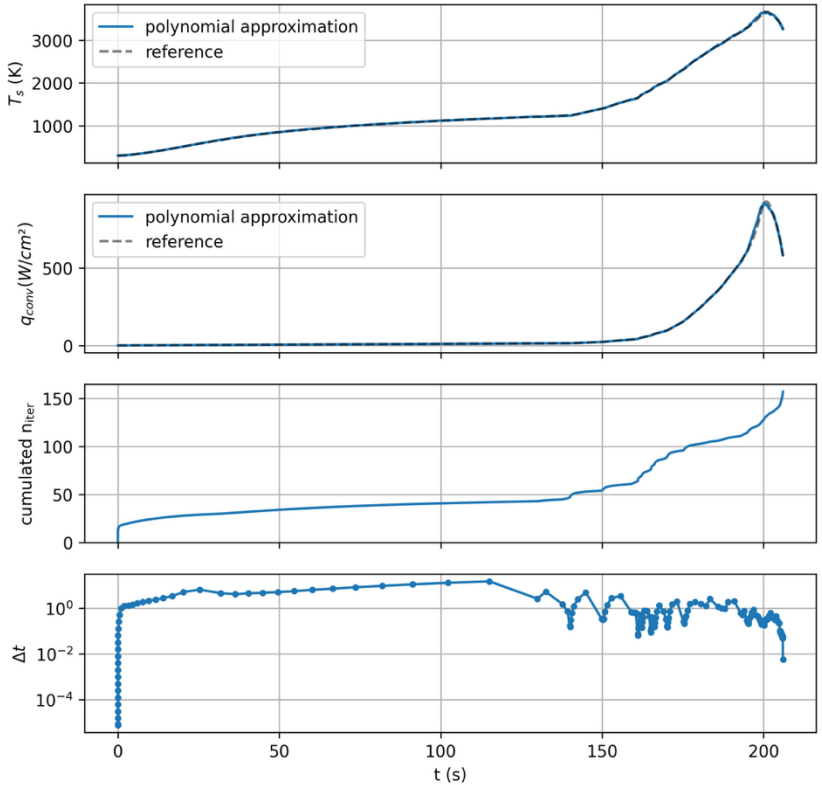


3rd-order explicit coupling ($rtol = 10^{-2}$)

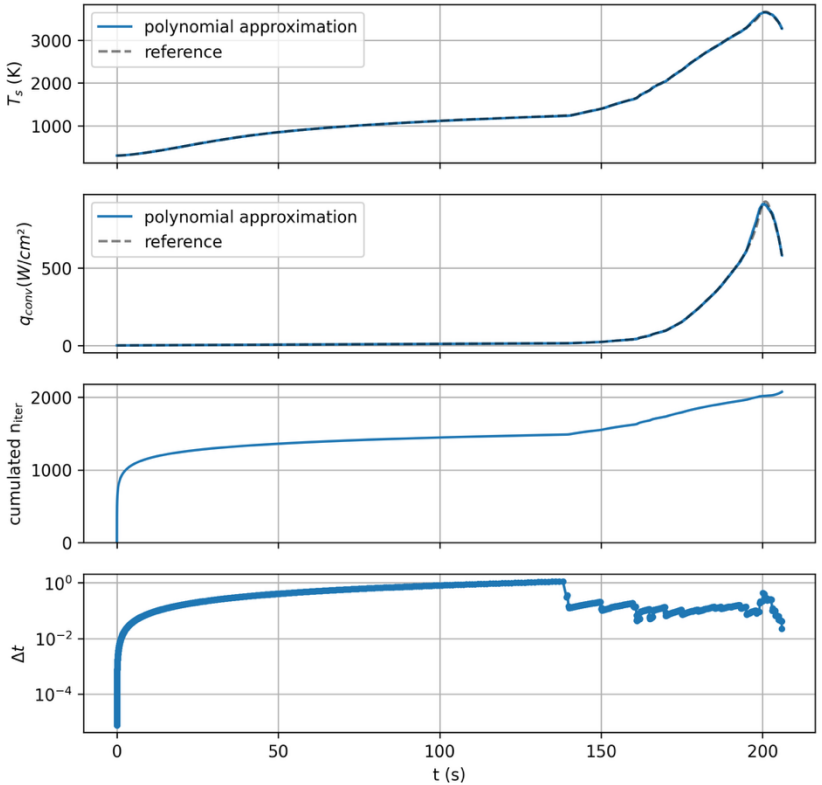


1st-order explicit coupling ($rtol = 3 \times 10^{-1}$)

Full entry trajectory

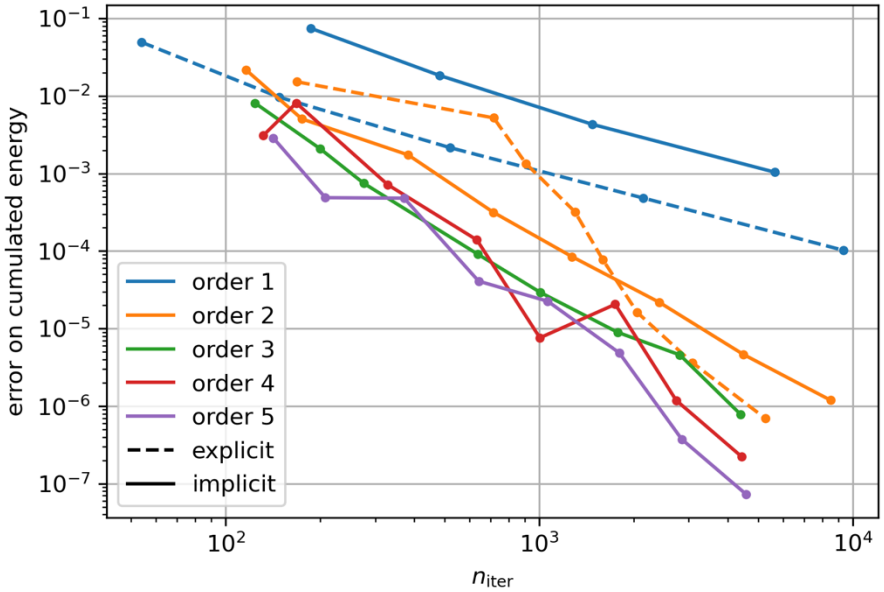


3rd-order explicit coupling ($rtol = 10^{-2}$)



1st-order explicit coupling ($rtol = 10^{-2}$)

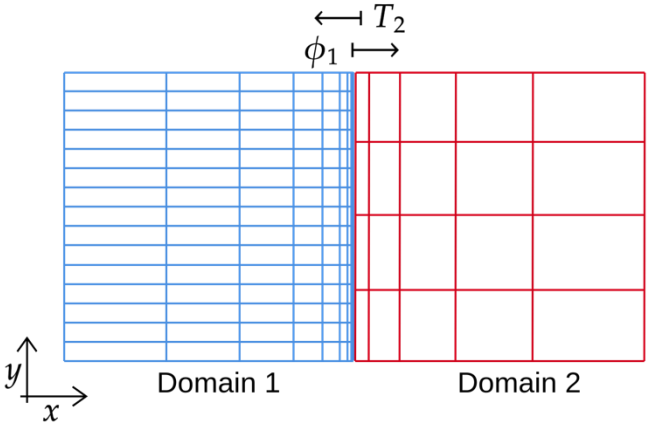
Full reentry trajectory – Work-precision diagram

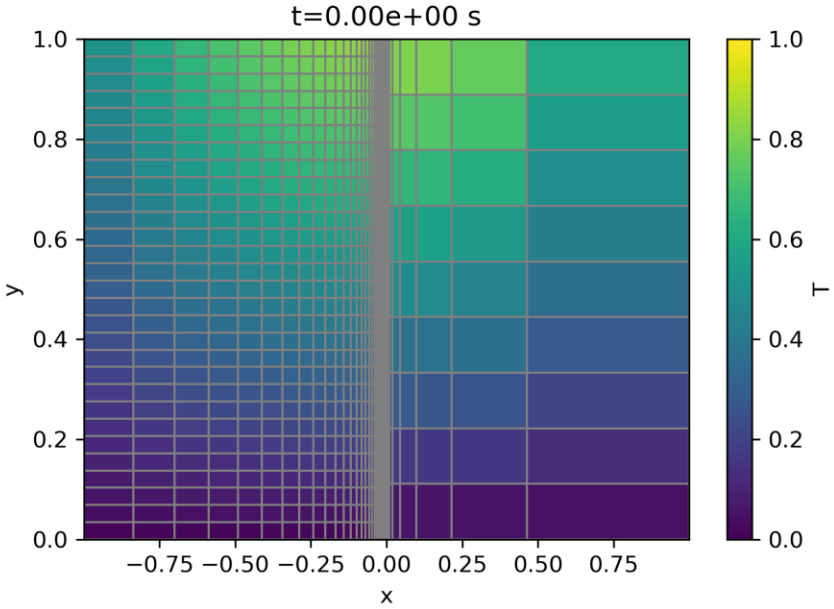


- Relative error on **energy absorbed** by the shield
$$\int_0^{210} \Phi_w(t)dt$$
- First-order explicit coupling has good stability, hence beats its implicit counterpart
→ **moderately stiff** coupling
- **Second-order** implicit easy to implement and accurate
- 3rd and higher-order explicit couplings unusable due to stability requirements
- **Higher-order implicit** couplings are most **efficient**

A case of stiff coupling

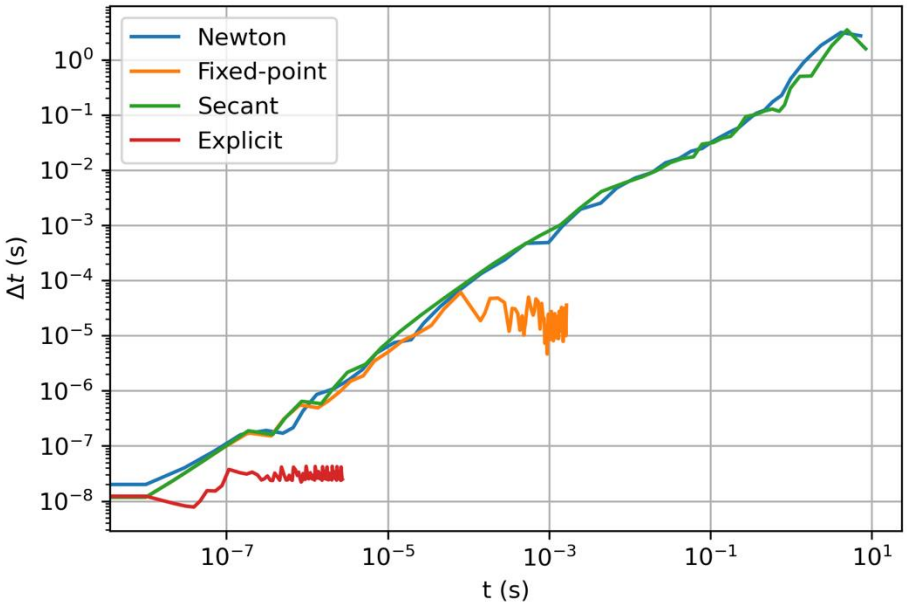
- 2D conjugate heat transfer
- Identical properties
- Dirichlet-Neumann coupling
- 3rd-order implicit coupling with various solvers:
  - Fixed-point
  - Damped Newton
  - Secant-like method (Jacobian-free)



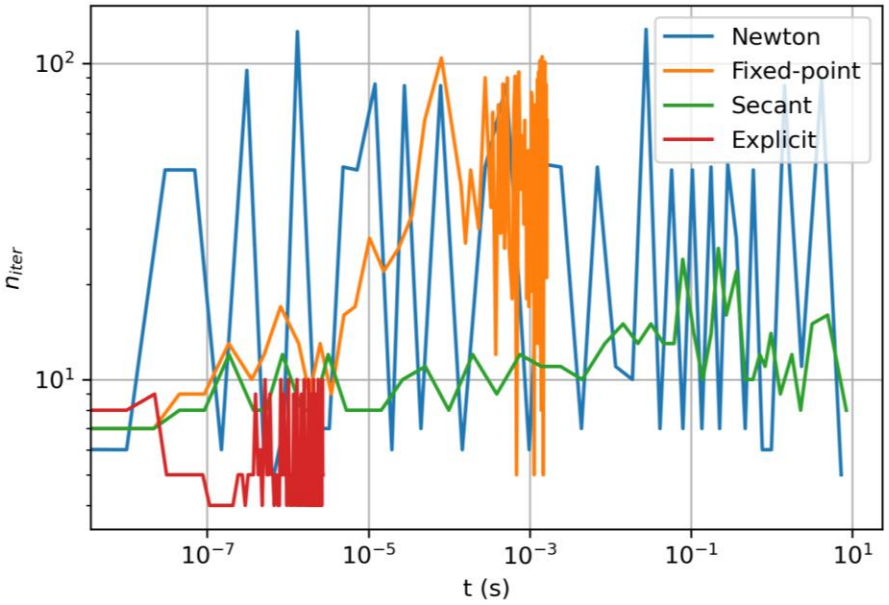


A case of stiff coupling

Time step evolution

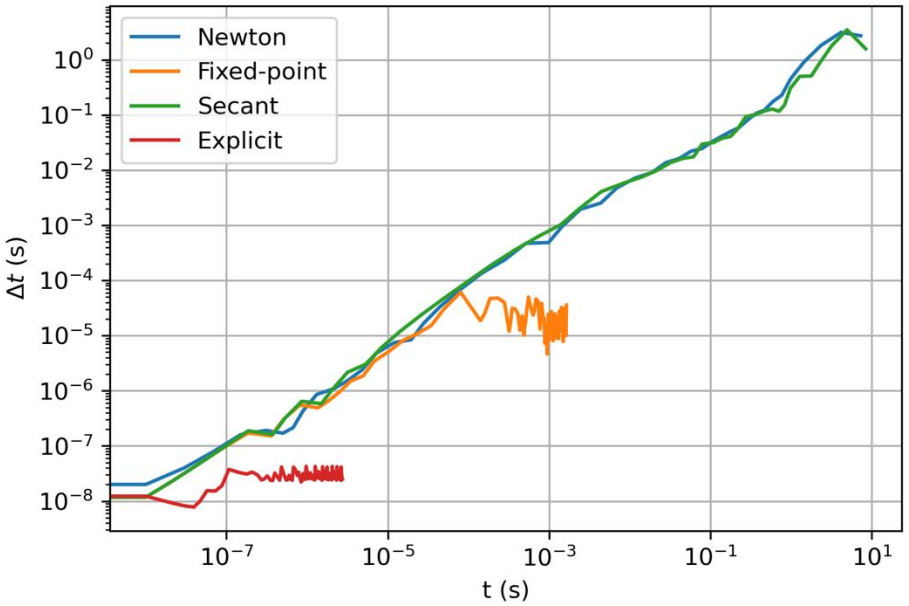


Cumulated number of iterations

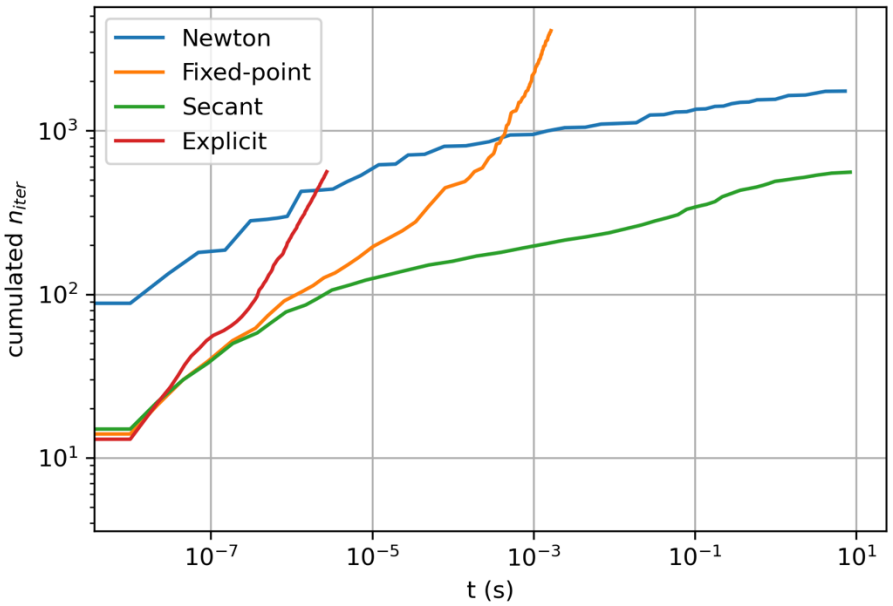


A case of stiff coupling

Time step evolution



Number of iterations per step



NASA Summer Program 2025

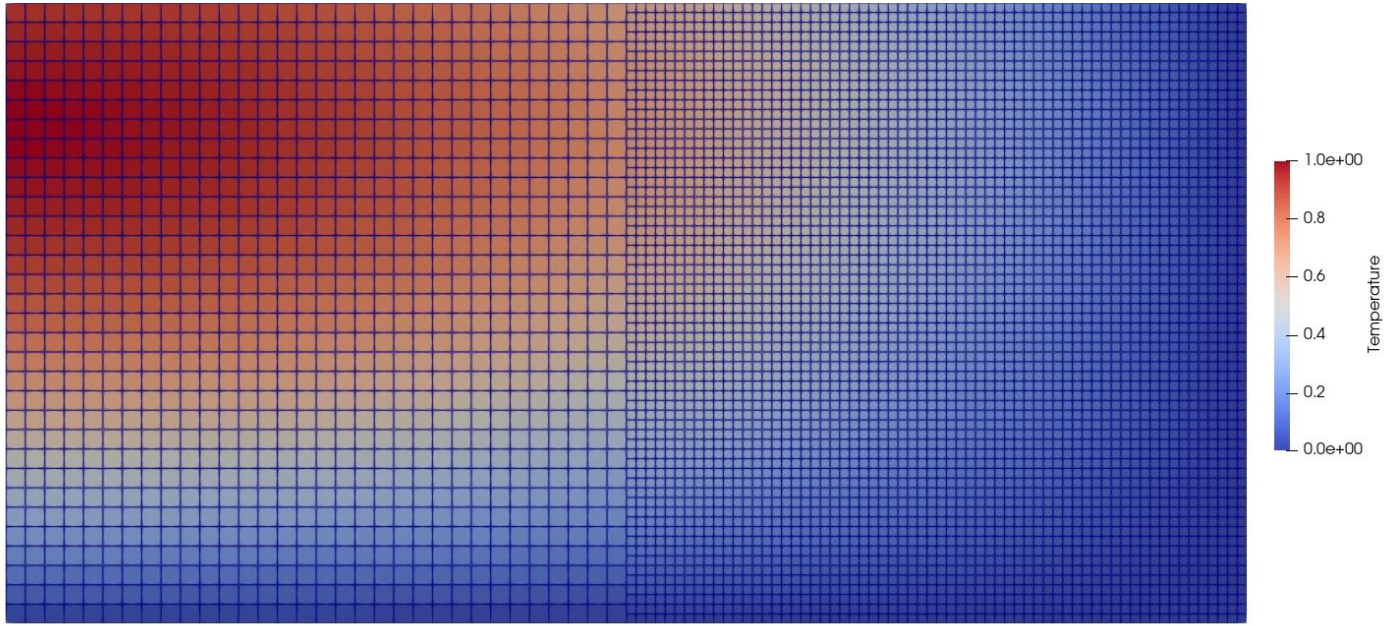


MODELING SUMMER VISIT 2025

- One month at NASA Ames, hosted by TSM (Bruno Dias)
- Work focused on the development of CWIPI 2.0
  - Successful implementation of the explicit adaptive multistep coupling
  - Implicit in the works

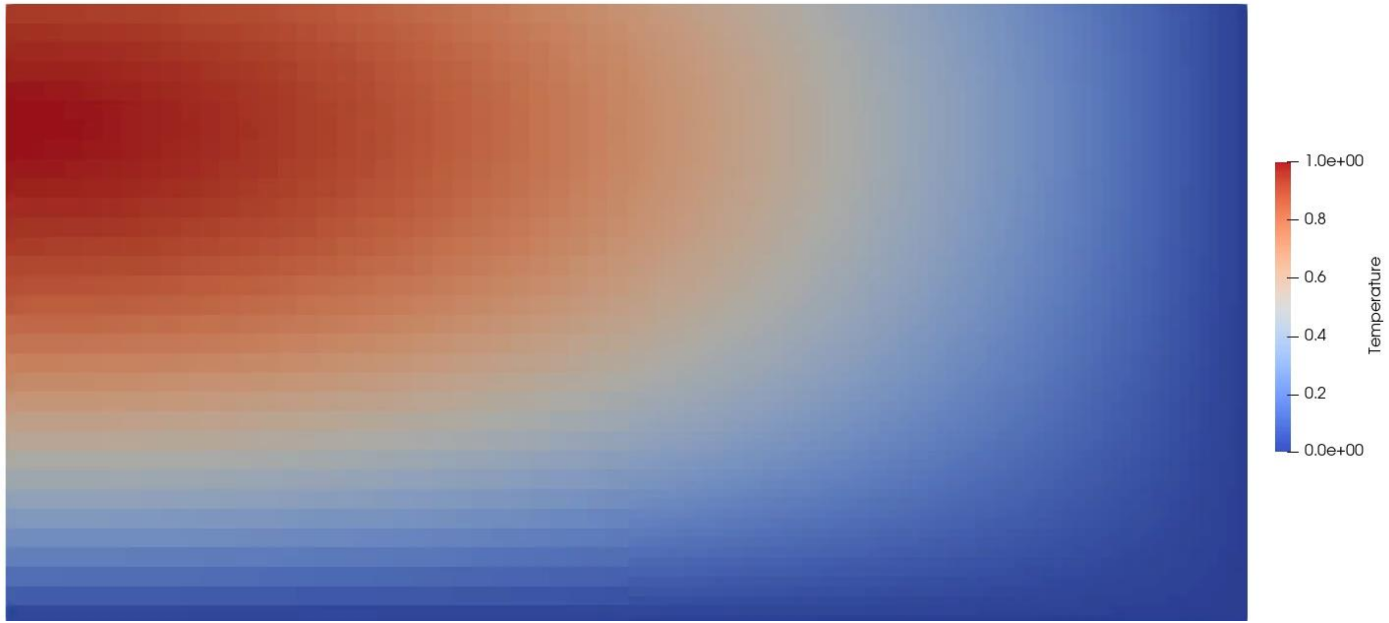
NASA Summer Program 2025

MODELING SUMMER VISIT 2025



NASA Summer Program 2025

MODELING SUMMER VISIT 2025



NASA Summer Program 2025 – CWIPI 2.0 API

```
CWP_clp_create("my-code" , "fluid-structure"  , "other-code") ) # declare coupling
CWP_Mesh_interf_vtx_set( ... ) # define interface mesh
CWP_create_field("WallTemperature", send=True) # this code will provide a wall temperature
CWP_create_field("HeatFlux", send=False) # and receive a heatflux

t = tn = t0
while tn < tf: # time loop
    tnp1 = CWP_getNextCouplingTime() # next coupling time point
    dt_CWP = tnp1 - tn # coupling time step

    dt = min( dt_CWP, solver_getPreferredTimeStep() )
    mycode_solve(t, dt) # single substep, calls CWP_predict(t+...)
    t += dt

    if CWP_perform_exchange(t): # MPI exchange triggered at tnp1
        CWP_issend ("my-code" , "my-coupling", "WallTemperature" ) # Send wall temperature to the other code
        CWP_irecv  ("my-code" , "my-coupling", "HeatFlux" ) # Receive interpolated heat flux from the other code
        CWP_wait_issend()
        CWP_wait_irecv()

        if CWP_validate_step(): # step is converged and accurate
            tn = tnp1
            mycode_store_state(tn)
        else: # go back and retry
            t = tn
            mycode_restoreState(tn)
```

Conclusions

- High-order multistep scheme promising:
  - **generalisation** of basic coupling
  - **Minimally intrusive**
  - **arbitrary order** of convergence in time
  - **dynamic time step** adaptation

- **Explicit** may be sufficient for **non-stiff** couplings

- **Implicit** is however required for stiffer configurations.

- Adaptation and high-order ensures **efficiency** and **accuracy**.

How can I try it ?

- Contact me ☺ at *laurent.francois@onera.fr*

- Open-source Python demonstrator **Rhapsopy** *github.com/hpc-maths/rhapsopy*
  - Many functionalities and tutorials
  - Easy prototyping
  - Non-HPC…
  - Requires a Python wrapper to your code

- **CWIPI 2.0** coming this year with the temporal API ! *github.com/onera/cwipi*

Perspectives

- Ongoing **3-year internal research project at** ONERA :
  - **HPC** implementation in CWIPI
  - Construct an **easy-to-use API** to leverage the multistep scheme
  - Revisit existing application cases

- **PhD** of A. Simon (co-funded with **Numpex**, the French Exascale program):
  - **Theory**
  - Efficient **HPC implicit algorithms** for CWIPI
  - Applications on Yales2/Modethec for **fire-induced material degradation**

- **Open for other collaborations ☺**

Thank you !

Acknowledgements:

- Bruno Dias and the whole Summer Program team
- Antoine Simon *(work on the multistep coupling + Rhapsopy + CWIPI)*
- Marc Massot *(work on the multistep coupling)*
- Ali Asad, Pierre Matalon *(Samurai CHT demonstrator)*
- Eric Quémerais and colleagues *(CWIPI 1.0 and related content)*
- Joël Dupays and Philippe Villedieu *(ONERA funding)*