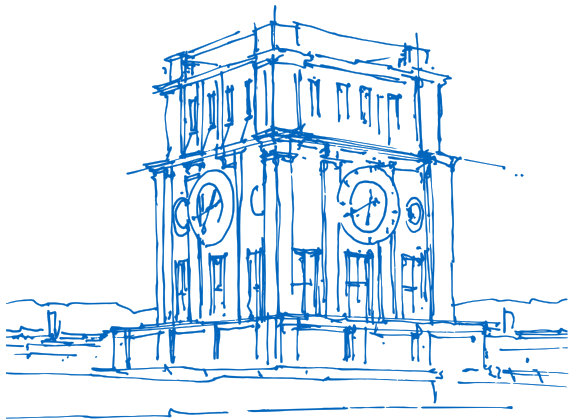


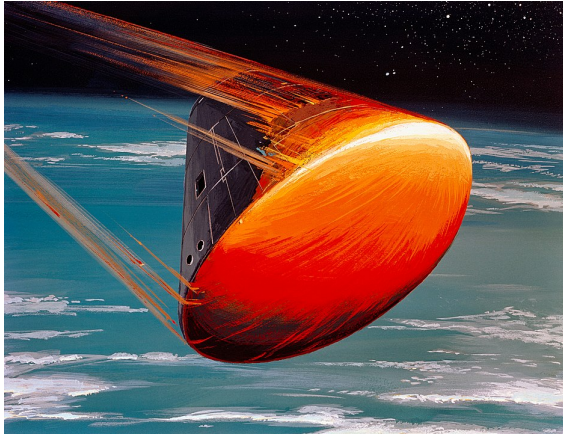
How preCICE Allows to Combine Heterogeneous Software, Discretization Techniques, and Modelling Approaches

**Benjamin Rodenberg, Benjamin
Uekermann, Hans-Joachim Bungartz**

COUPLED 2025, May 27, 2025



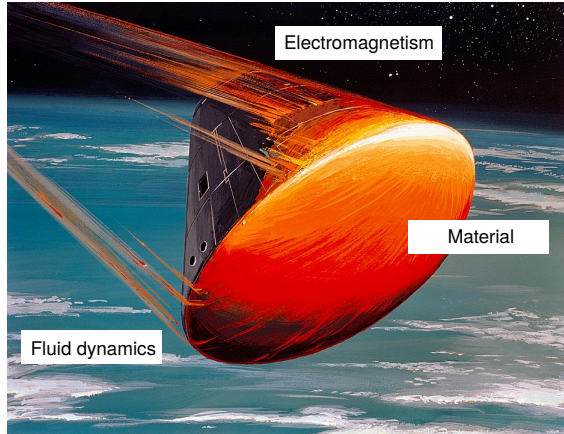
Multiphysics application: thermal protection system (1)



By North American Rockwell, Public Domain,

<https://commons.wikimedia.org/w/index.php?curid=2466251>

Multiphysics application: thermal protection system (1)

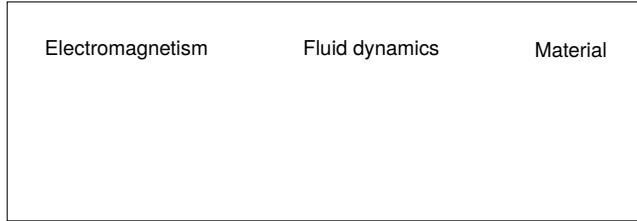


By North American Rockwell, Public Domain,

<https://commons.wikimedia.org/w/index.php?curid=2466251>

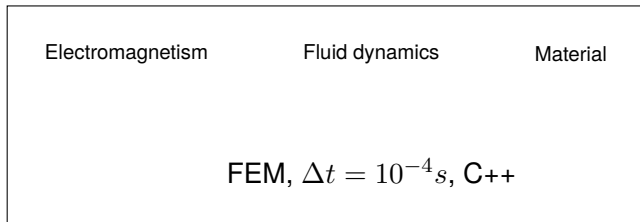
Multiphysics application: thermal protection system (2)

Monolithic multiphysics software



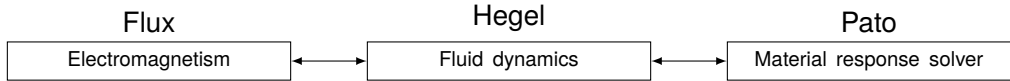
Multiphysics application: thermal protection system (2)

Monolithic multiphysics software



- **Space discretization** with FEM
- **Global** time step size $\Delta t = 10^{-4}s$
- **Programming language** C++

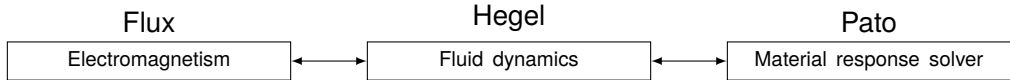
Multiphysics application: thermal protection system (3)



Setup for partitioned simulation of inductively coupled plasma wind tunnels (skipping Plato solver)¹

¹Munafò, A., Chiodi, R., Kumar, S., Le Maout, V., Stephani, K.A., Panerai, F., Bodony, D.J. and Panesi, M., 2022. A multi-physics modeling framework for inductively coupled plasma wind tunnels. In AIAA SCITECH 2022 Forum (p. 1011).

Multiphysics application: thermal protection system (3)



Setup for partitioned simulation of inductively coupled plasma wind tunnels (skipping Plato solver)¹

Goal: Digital Twins ← Multiphysics / Multi-science

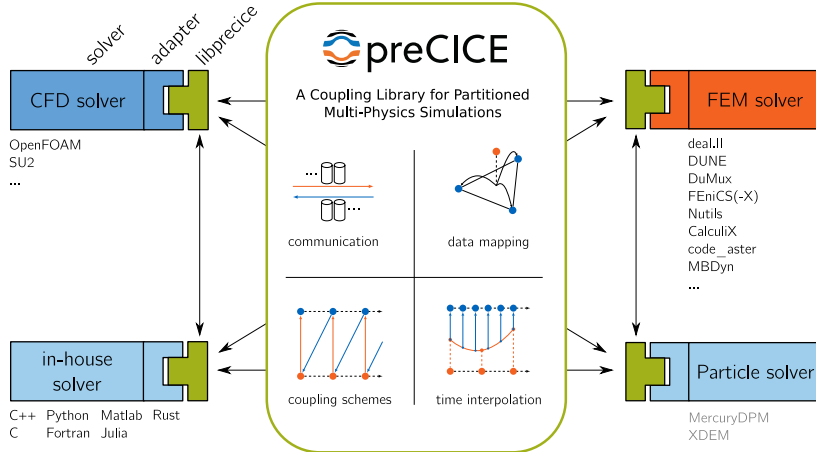
- Multi-language: software & programming languages
- Multi-scale: resolutions in space & time
- Multi-model: FEM, PINN, DAEs, ...

¹Munafò, A., Chiodi, R., Kumar, S., Le Maout, V., Stephani, K.A., Panerai, F., Bodony, D.J. and Panesi, M., 2022. A multi-physics modeling framework for inductively coupled plasma wind tunnels. In AIAA SCITECH 2022 Forum (p. 1011).

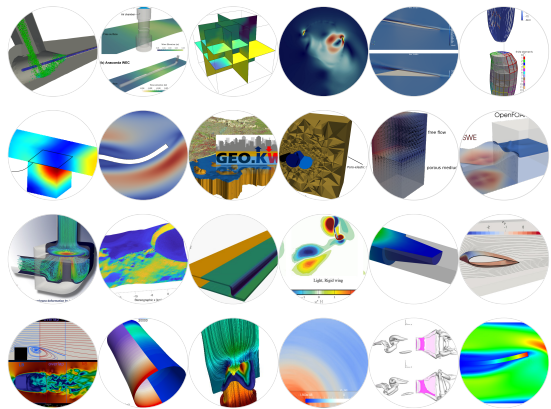
Outline

- 1 Multiphysics application: thermal protection system
 - tutorials/perpendicular-flap
 - User interface
- 2 Multi-science for Digital Twins
 - Multi-language
 - Multi-solver
 - Multi-scale
 - Multi-model
- 3 Conclusion

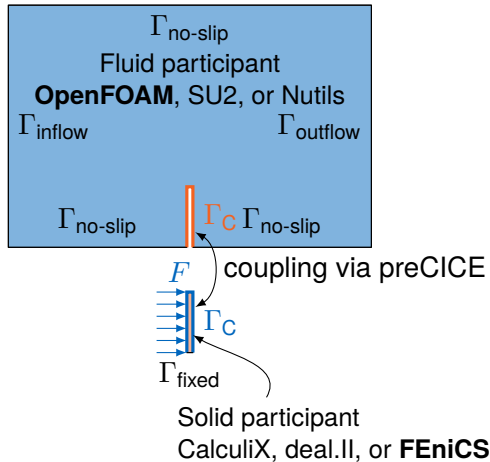
The coupling library preCICE¹



¹Chourdakis, G., Davis, K., Rodenberg, B., Schulte, M., Simonis, F., Uekermann, B., Abrams, G., et al. 2022. preCICE v2: A sustainable and user-friendly coupling library. Open Research Europe, 2, p.51.



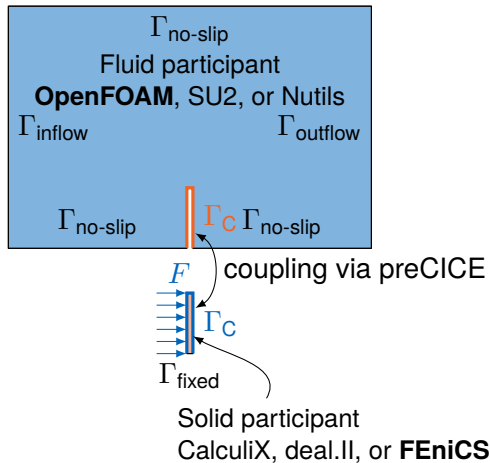
DEMO TIME



Divide

Conquer

Combine

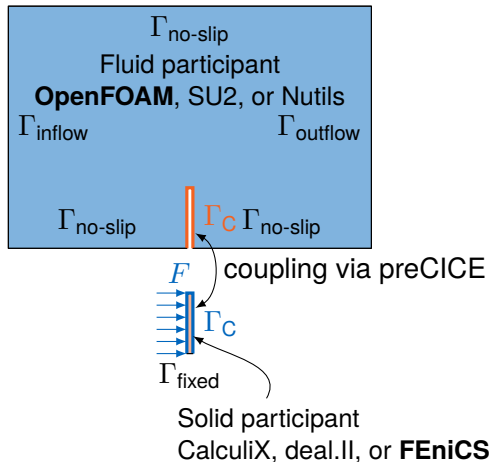


Divide

- OpenFOAM $\neq$ FEniCS
- preCICE = *partitioned* multiphysics
- Dirichlet-Neumann = black box

Conquer

Combine



Divide

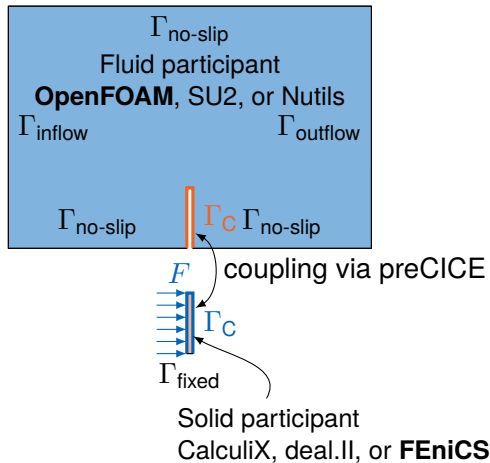
Conquer

- Fluid: $\mathcal{F}(d) = f$
- Solid: $\mathcal{S}(f) = d$

Boundary response maps

(= Poincaré-Steklov operator)

Combine



Divide

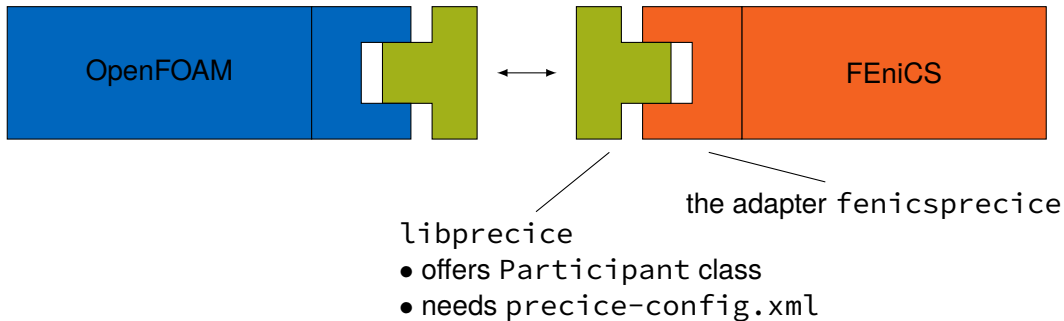
Conquer

Combine

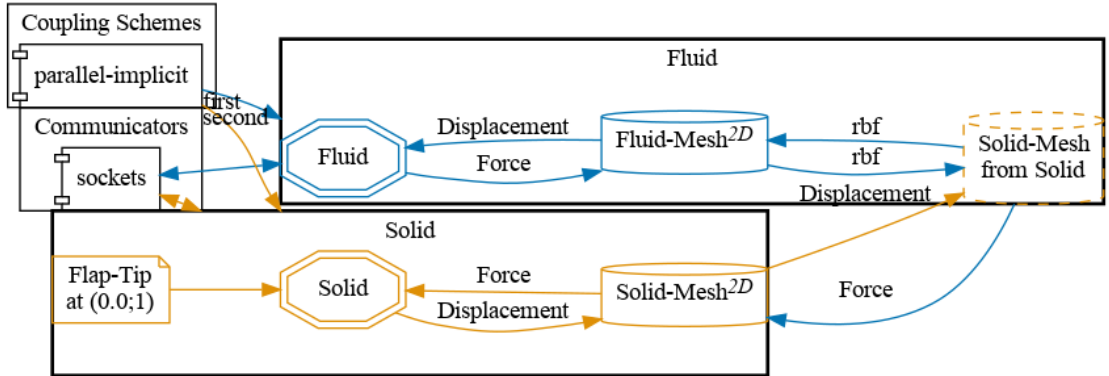
- $\mathcal{F}(\mathcal{S}(f^k)) = \tilde{f}^k$
- $\tilde{f}^k \xrightarrow{\mathcal{A}} f^{k+1}$

Picard iteration + acceleration

DEMO TIME



precice-config.xml (1)



From precice-config-visualizer

precice-config.xml (2)

```

<precice-configuration>
  <!-- declare  $f$  and  $d$  -->
  <data:vector name="Force" />
  <data:vector name="Displacement" />
  <mesh name="Solid-Mesh" dimensions="2" />
  <mesh name="Fluid-Mesh" dimensions="2" />
  <!--  $S(f) = d$  -->
  <participant name="Solid">
    <provide-mesh name="Solid-Mesh" />
    <write-data name="Displacement" mesh="Solid-Mesh" />
    <read-data name="Force" mesh="Solid-Mesh" />
  </participant>
  <!--  $\mathcal{F}(d) = f$  -->
  <participant name="Fluid">
    ...
    <mapping:rbf direction="write" from="Fluid-Mesh" to="Solid-Mesh" />
    <mapping:rbf direction="read" from="Solid-Mesh" to="Fluid-Mesh" />
  </participant>
  ...
</precice-configuration>

```

precice-config.xml (3)

```

<precice-configuration>
...
<m2n:sockets acceptor="Fluid" connector="Solid" exchange-directory=".." />
<!--  $S(\mathcal{F}(d^k)) = \tilde{d}^{k+1}$  -->
<coupling-scheme:serial-implicit>
  <time-window-size value="0.01" />
  <max-time value="5" />
  <participants first="Fluid" second="Solid" />
  <exchange data="Force" mesh="Solid-Mesh" from="Fluid" to="Solid" />
  <exchange data="Displacement" mesh="Solid-Mesh" from="Solid" to="Fluid" />
  <max-iterations value="50" />
  <relative-convergence-measure limit="5e-3" data="Displacement" mesh="Solid-Mesh" />
  <relative-convergence-measure limit="5e-3" data="Force" mesh="Solid-Mesh" />
  <!--  $\tilde{f}^k \xrightarrow{A} f^{k+1}$  -->
  <acceleration:IQN-ILS>
    ...
  </acceleration:IQN-ILS>

</coupling-scheme:parallel-implicit>
</precice-configuration>

```

Using the Participant class

```
# example: FSI coupling, perspective of solid solver S:
# ...
participant = Participant("Solid", "precice-config.xml")

ids = participant.setMeshVertices("Solid-Mesh", coords)
participant.initialize()

while participant.is_coupling_ongoing():
    # store checkpoint, if needed
    dt = participant.get_max_time_step_size()
    forces = participant.read_data("Solid-Mesh", "Force", ids, dt)
    displ = time_stepper.do_step(forces, dt) #  $d_{n+1} = d_n + \Delta t \dots$ 
    participant.write_data("Solid-Mesh", "Displacement", ids, displ)
    participant.advance(dt)
    # read checkpoint, if needed

participant.finalize()
```

Example: Mesh initialization

```
mesh = fenics.RectangleMesh((-1,0),(1,20),4,26)
V = fenics.VectorFunctionSpace(mesh, 'P', 2)
```

Example: Mesh initialization

```
mesh = fenics.RectangleMesh((-1,0),(1,20),4,26)
V = fenics.VectorFunctionSpace(mesh, 'P', 2)

# how to get coords from mesh or V?
participant.set_mesh_vertices("Solid-Mesh", coords)
```

Example: Mesh initialization

```
mesh = fenics.RectangleMesh((-1,0),(1,20),4,26)
V = fenics.VectorFunctionSpace(mesh, 'P', 2)
```

how to get coords from mesh or V?

```
participant.set_mesh_vertices("Solid-Mesh", coords)
```



put boilerplate into adapter:

```
adapter = Adapter(config="adapter-config.json")
adapter.initialize(read_function_space=V, write_object=V)
```

Example: Read data

```
fenics.solve(A, x, b)
```

Example: Read data

```
forces = participant.read_data("Solid-Mesh", "Force")  
# how to apply forces to FEM system?
```

```
fenics.solve(A, x, b)
```

Example: Read data

```
forces = participant.read_data("Solid-Mesh", "Force")  
# how to apply forces to FEM system?
```



```
read_data = adapter.read_data()  
# convert to FEniCS data structure  
b = adapter.get_point_sources(read_data)  
# or:  
adapter.update_coupling_expression(read_data)  
fenics.solve(A, x, b)
```

Outline

- 1 Multiphysics application: thermal protection system
 - tutorials/perpendicular-flap
 - User interface
- 2 Multi-science for Digital Twins
 - Multi-language
 - Multi-solver
 - Multi-scale
 - Multi-model
- 3 Conclusion

Monolithic multiphysics software

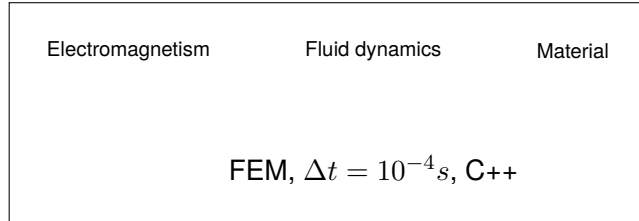
Electromagnetism

Fluid dynamics

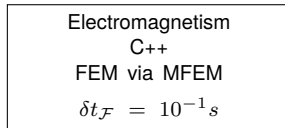
Material

FEM, $\Delta t = 10^{-4}s$, C++

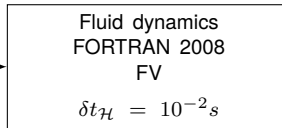
Monolithic multiphysics software



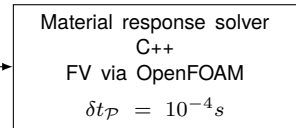
Flux



Hegel



Pato



Multi-language

C++, C, FORTRAN

- preCICE is written in C++
- Native language bindings in C and FORTRAN

Python, MATLAB, Julia, Rust, FORTRAN module

- Non-native bindings
- Shipped & installed independently

5 repositories

...  

fortran-module Public

A Fortran module for the Fortran bindings of preCICE (formerly "f2003 bindings")

Fortran •  GNU Lesser General Public License v3.0 •  6 •  5 •  4 •  3 • Updated last week

python-bindings Public

Python language bindings for preCICE

Python •  GNU Lesser General Public License v3.0 •  16 •  24 •  30 •  2 • Updated 2 weeks ago

matlab-bindings Public

MATLAB language bindings for preCICE

MATLAB •  GNU Lesser General Public License v3.0 •  3 •  5 •  15 •  1 • Updated on Apr 4

PreCICE.jl Public

Julia language bindings for preCICE

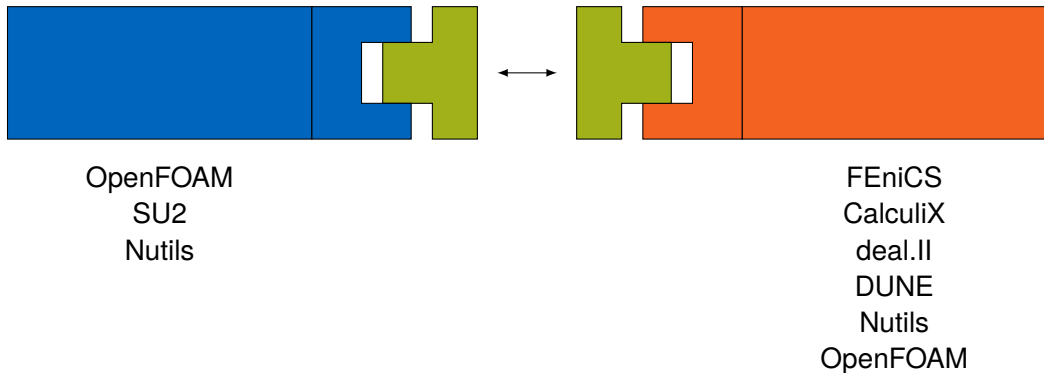
Julia •  GNU Lesser General Public License v3.0 •  2 •  13 •  6 •  1 • Updated on Apr 2

rust-bindings Public

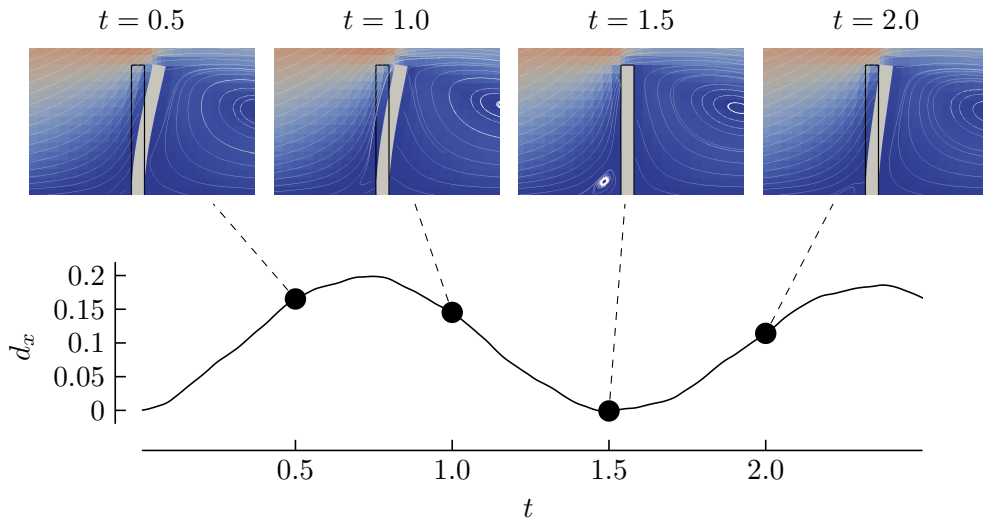
preCICE language bindings for rust

rust bindings

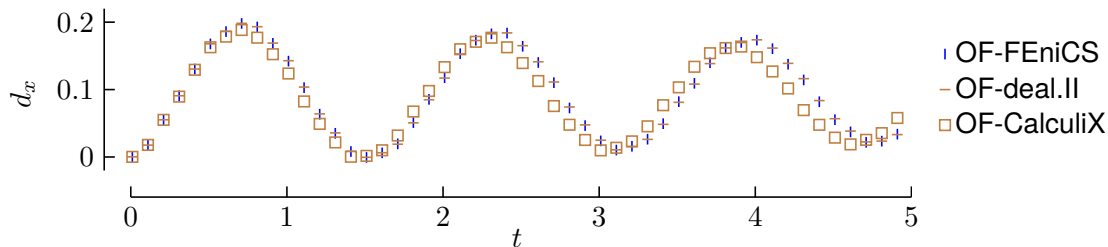
Rust •  GNU Lesser General Public License v3.0 •  1 •  2 •  0 •  0 • Updated on Apr 2



Real life $\neq$ analytical solutions (1)



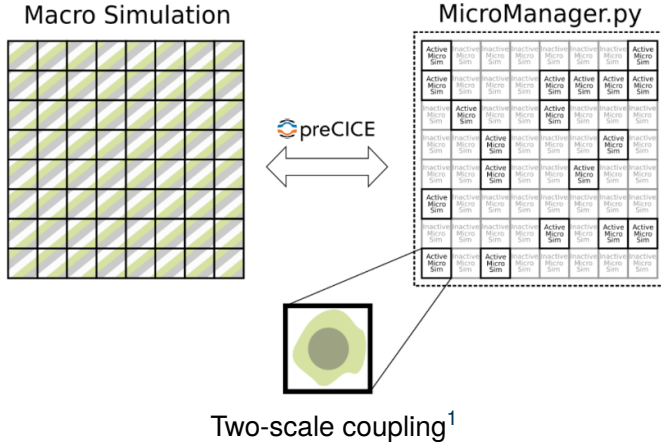
Real life $\neq$ analytical solutions (2)



Why this gap?

- It turned out that we used (locking) C3D8 elements in CalculiX
- Fixed by Andrés Pedemonte Fehrmann via <https://github.com/precice/tutorials/pull/250>

Multi-scale in space

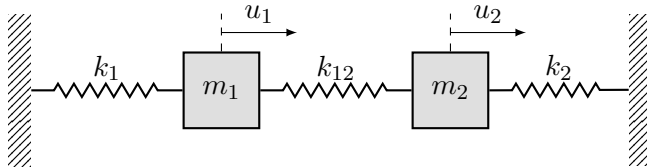


Talk at COUPLED

- Ishaan Desai:
An Application-Agnostic Software Solution for Model Adaptivity of Two-Scale Coupled Problems
- Wed 14:00 - 16:00
- Room: Sala delle Ceramiche

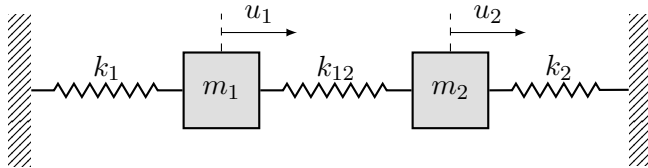
¹Desai, I., Scheurer, E., Bringedal, C. and Uekermann, B., 2023. Micro Manager: a Python package for adaptive and flexible two-scale coupling. *Journal of Open Source Software*, 8(91), p.5842.

Multi-scale in time¹(1)



¹Rodenberg, B., 2025. Flexible and robust time stepping for partitioned multiphysics. Dissertation [Unpublished manuscript]

Multi-scale in time¹(1)

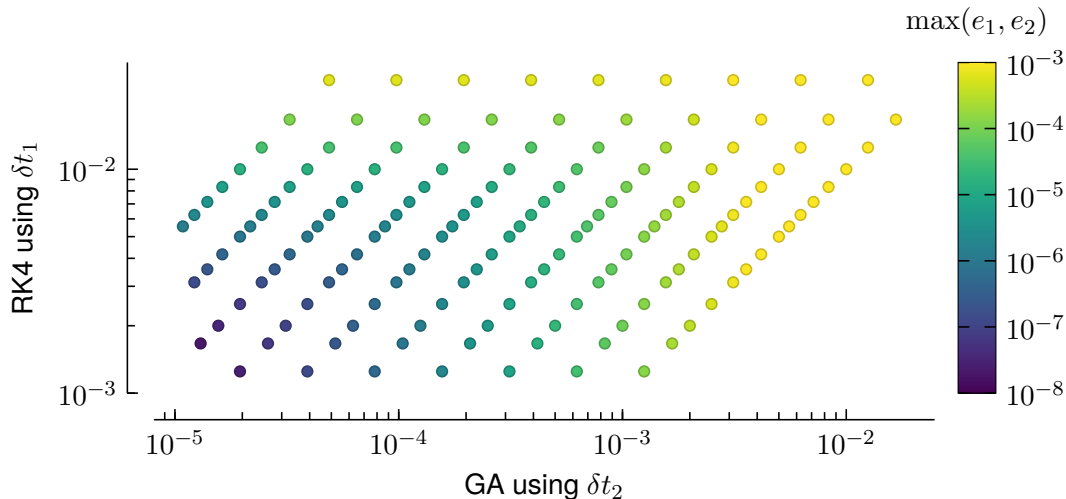


time interpolation since preCICE v3

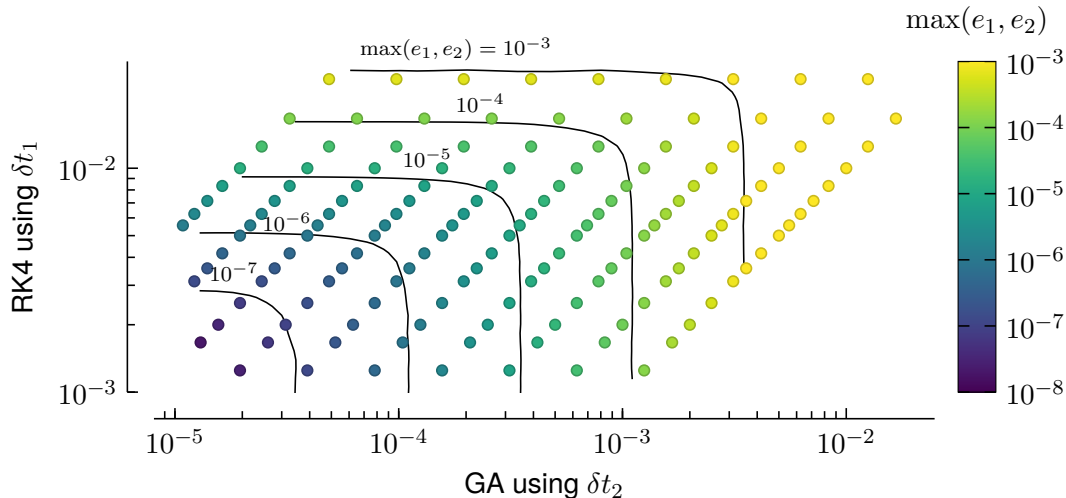
- $\Delta t > \delta t$ (subcycling), $\delta t_1 \neq \delta t_2$ (multirate)
- `read_data()` $\rightarrow$ `read_data(time)`
- quasi-Newton waveform iteration

¹Rodenberg, B., 2025. Flexible and robust time stepping for partitioned multiphysics. Dissertation [Unpublished manuscript]

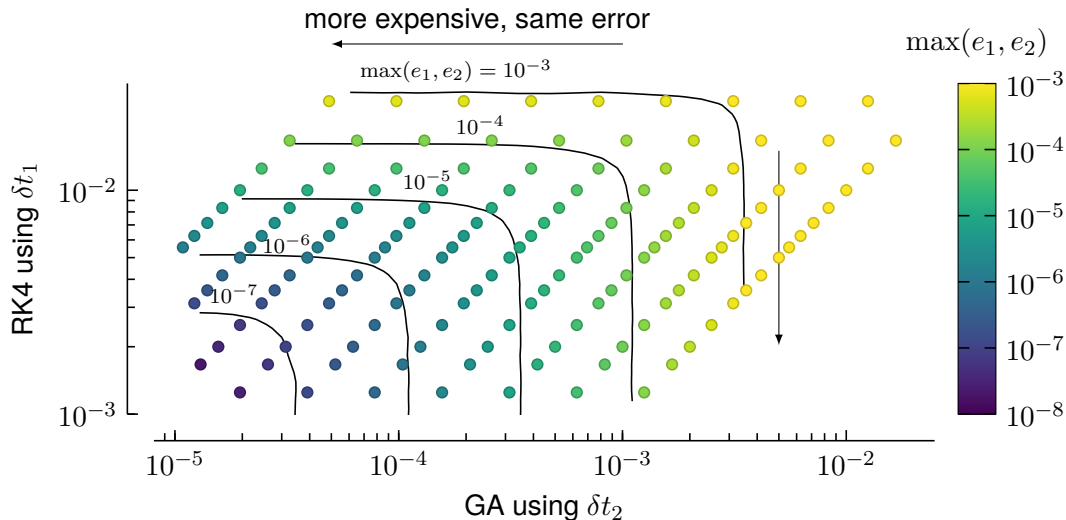
Multi-scale in time (2)



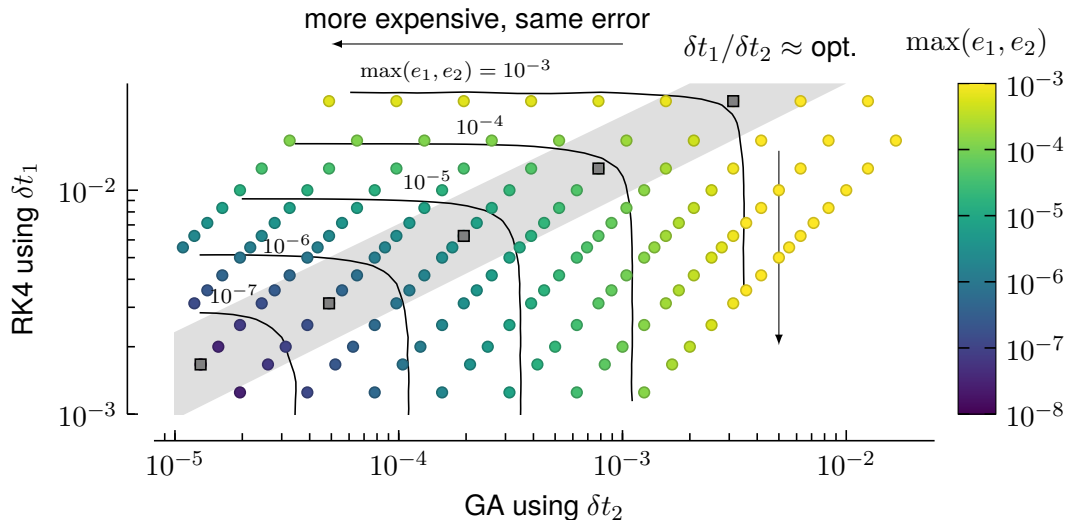
Multi-scale in time (2)

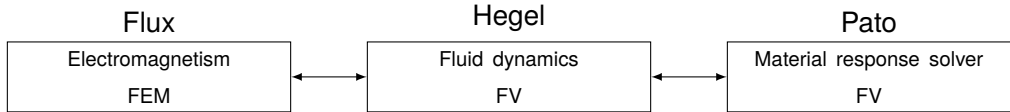


Multi-scale in time (2)

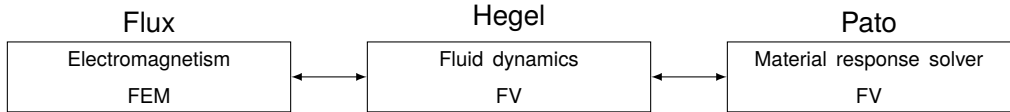


Multi-scale in time (2)





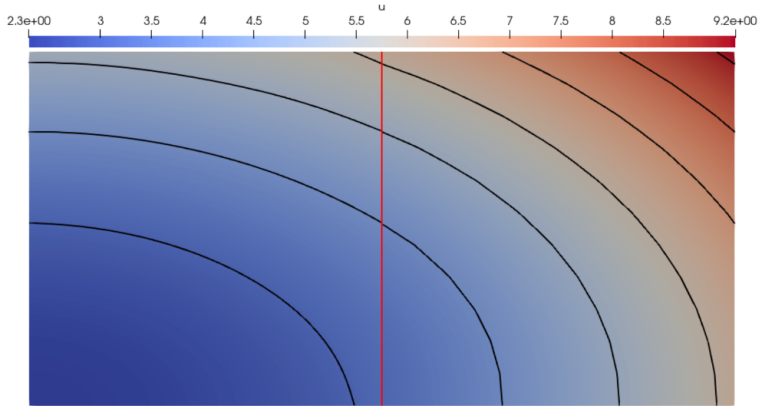
¹Willeke, L., Schneider, D. and Uekermann, B., 2023. A preCICE-FMI Runner to Couple FMUs to PDE-Based Simulations. In Modelica Conferences (pp. 479-486).



Also for no Finite-X approaches?

- Modelica, Simulink, FMI/FMU¹
- Physics-informed neural networks (PINN)

¹Willeke, L., Schneider, D. and Uekermann, B., 2023. A preCICE-FMI Runner to Couple FMUs to PDE-Based Simulations. In Modelica Conferences (pp. 479-486).



Dirichlet-Neumann coupling for partitioned heat equation (Nvidia Modulus/FEniCS)¹

¹Klimont, D., 2025. Coupling physics-informed neural networks from Nvidia Modulus with conventional models using preCICE. Bachelor thesis.

How it works (prototype!)

Dirichlet solver:

$$\mathcal{D}(u) = q$$

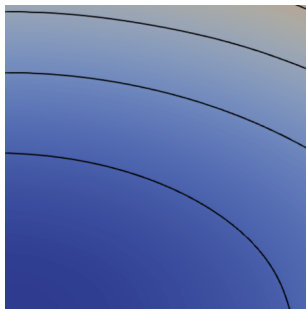
Solved using Nvidia Modulus and PINN.

Tricky part:

$$u(x, y, t) \quad \forall y \in \Gamma, t \in [0, T]$$

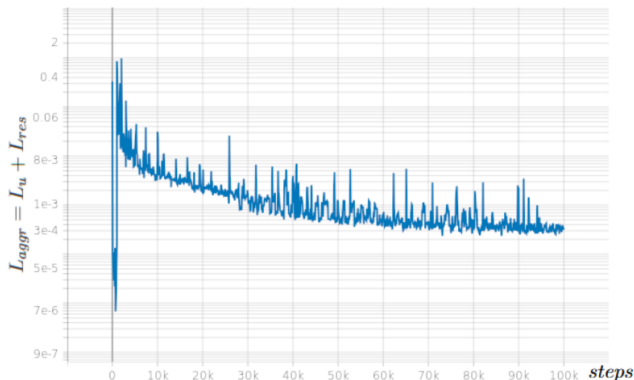
Picard iteration & accelerate:

$$\mathcal{N}(\mathcal{D}(u)) = u$$



u on Γ from $\mathcal{N}$

Dirichlet solver (PINN)



Open points

- Runtime
- $\Delta t = T$
- Optimize in early coupling iterations?

Loss; preCICE performs coupling iteration every 1k steps.¹

¹Klimont, D., 2025. Coupling physics-informed neural networks from Nvidia Modulus with conventional models using preCICE. Bachelor thesis.

Couple your code with

CalculiX

deal.II

DuMux

DUNE

FEniCS

Nutils

OpenFOAM

SU2

...

for partitioned
multi-physics
simulations



preCICE

Workshop #6

2025

HSU Hamburg

September

Germany

8 - 12

precice.org

Training
Talks
Posters
Support
Community



preCICE

- perpendicular flap
- v3 API and config
- time interpolation feature complete
- adapters always welcome

Digital Twins

- Multi-language
- Multi-solver
- Multi-scale (looking for adopters!)
- Multi-model (FMI, SPICE, PINN)
- Multi-X (anything missing?)