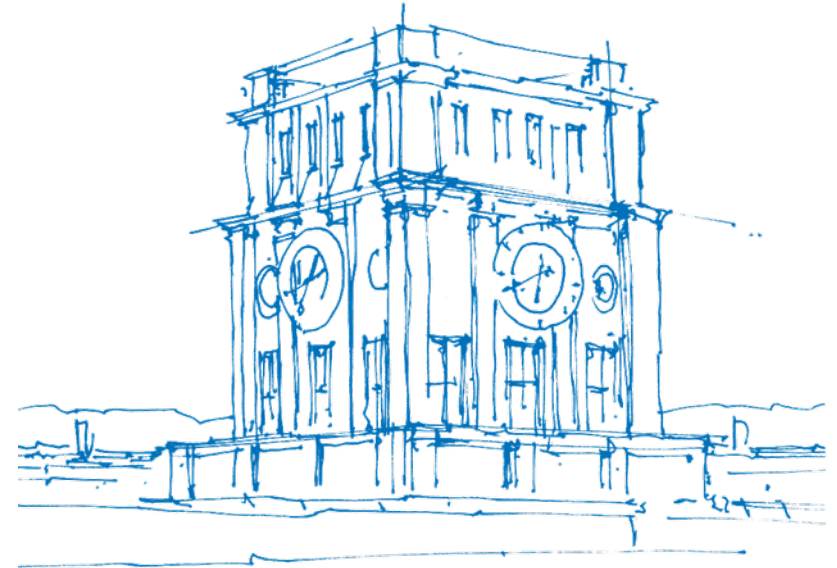


# Waveform iteration in partitioned multiphysics with preCICE

Benjamin Rodenberg, B. Uekermann, M. Schulte, H.J. Bungartz

PinT 2024, Feb. 6, 2024



*TUM Uhrenturm*

# Outline



preCICE overview

tutorials/perpendicular-flap

preCICE v3: Time interpolation

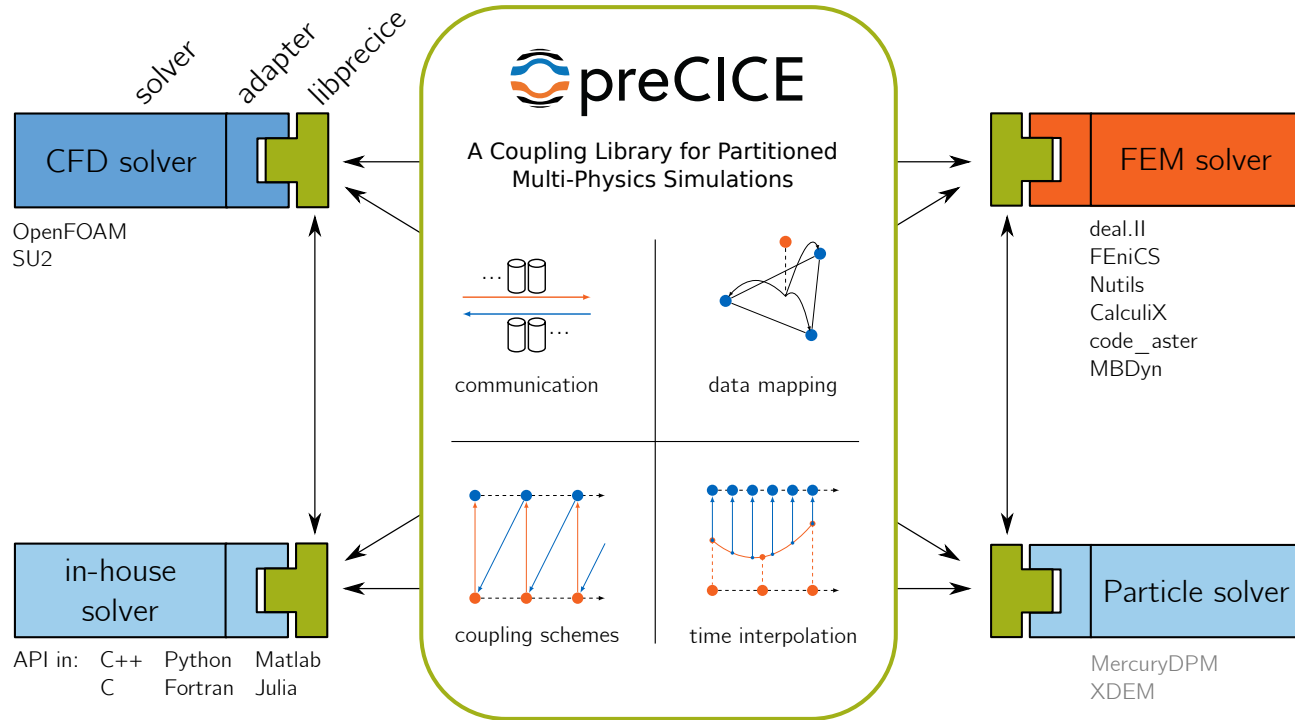
tutorials/oscillator

tutorials/partitioned-heat-conduction

Conclusion



# preCICE overview



# github.com/precice/tutorials



precice / tutorials

Code Issues (57) Pull requests (17) Discussions Actions Projects Security Insights Settings

tutorials (Public) Sponsor Edit Pins Watch (11) Fork (98) Starred (95)

develop 21 Branches 6 Tags Go to file Add file Code

This branch is 118 commits ahead of, 1 commit behind master . Contribute

IshaanDesai Remove call to initialize() from Micro Manager run scripts ✓ 89e36d2 · 2 days ago 692 Commits
multiple-perpendicular-flaps Use endTime like max-time in precice-config.xml (#451) 5 days ago
oscillator solve compatibility issue from pandas (#452) 5 days ago
partitioned-backwards-facing-step Use endTime like max-time in precice-config.xml (#451) 5 days ago
partitioned-elastic-beam Move dimensions to each mesh (#367) 6 months ago
partitioned-heat-conduction-complex Format our config files according to v3 2 months ago
partitioned-heat-conduction-direct Add nutils requirements (#439) last week
partitioned-heat-conduction Add nutils requirements (#439) last week
partitioned-pipe-two-phase Use endTime like max-time in precice-config.xml (#451) 5 days ago
partitioned-pipe Use endTime like max-time in precice-config.xml (#451) 5 days ago
perpendicular-flap Use endTime like max-time in precice-config.xml (#451) 5 days ago
quickstart Port quickstart to precice v3 (#414) 2 weeks ago
tools Add DuMuX as a participant to the two-scale-heat-condu... last week
turek-hron-fsi3 Format our config files according to v3 2 months ago

About

Various tutorial cases for the coupling library preCICE with real solvers. These files are meant to be rendered on precice.org, so don't look at the README files here.

[www.precice.org/](http://www.precice.org/)

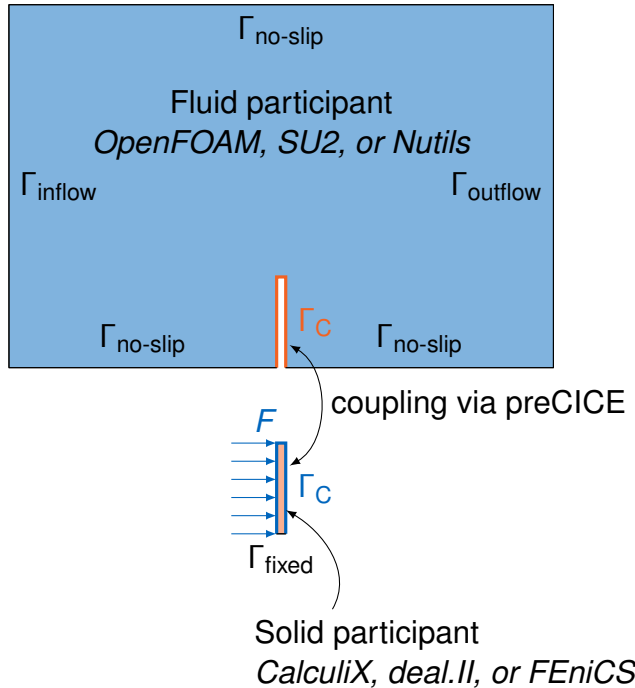
tutorial openfoam multiphysics coupling calculus multi-physics fluid-structure-interaction conjugate-heat-transfer su2 precice

Readme LGPL-3.0 license Code of conduct Activity Custom properties 95 stars 11 watching 98 forks Report repository

Releases (4)

v202211.0 - Before the iceber... Latest on Nov 22, 2022

DEMO TIME



## Divide

- OpenFOAM  $\neq$  FEniCS
- Dirichlet-Neumann = black box

## Conquer

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

## Boundary response maps

(= Poincaré-Steklov operator)

## Combine

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

Picard iteration + acceleration

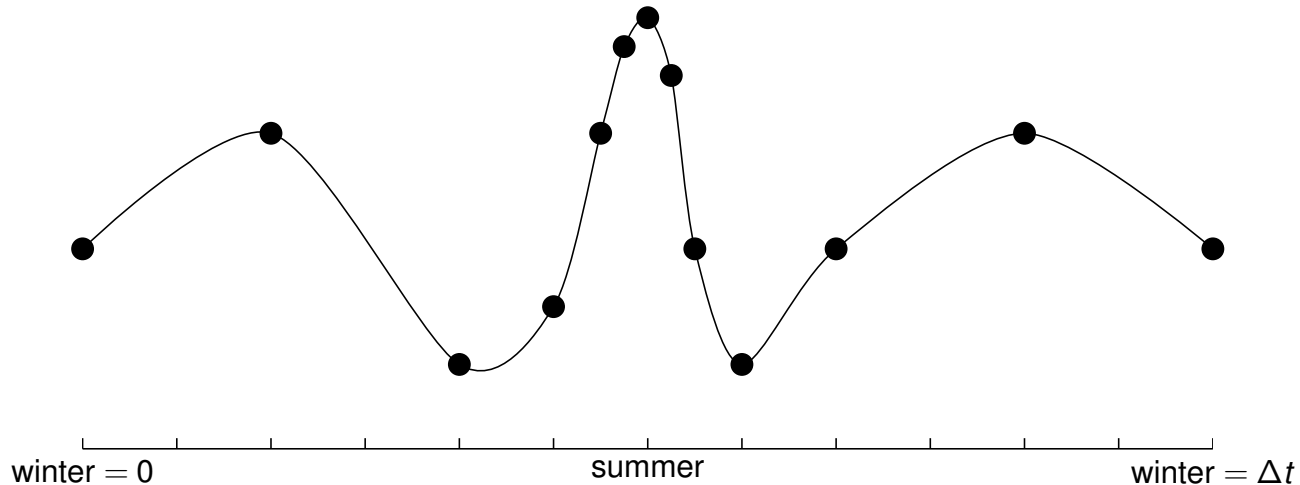
DEMO TIME



# preCICE v3: Time interpolation

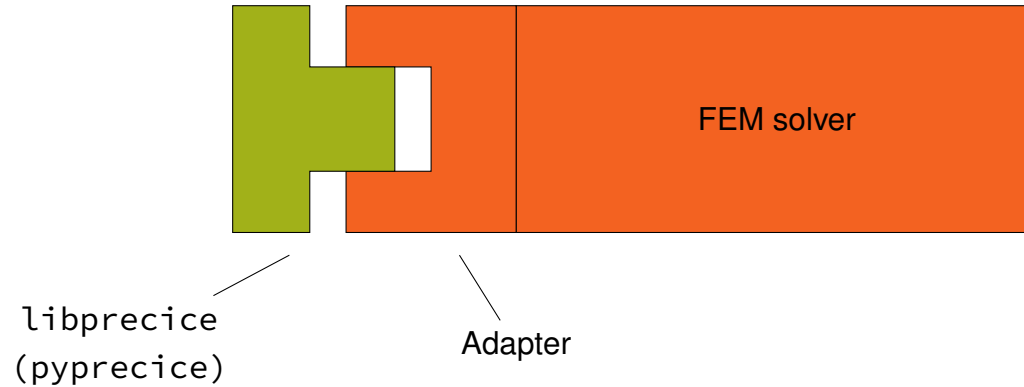
## preCICE Workshop 2023: Two "ice sheet talks"<sup>1</sup>

- Yannic Fischler: *A preCICE-interface for the ice-sheet and sea-level system model*
- Daniel Abele: *Coupling an ice sheet model with satellite image based simulation of calving fronts*



<sup>1</sup><https://precice.org/precice-workshop-2023.html>

# Requirements for time interpolation in preCICE



## Goal: Black-box + higher-order + multirate

- Numerics: Accuracy, convergence order, energy conservation
- API Design: Simple user interface!
- Move as much multirate/higher-order/interpolation logic *inside* preCICE as possible
- preCICE v3 + new API was just released<sup>1</sup>

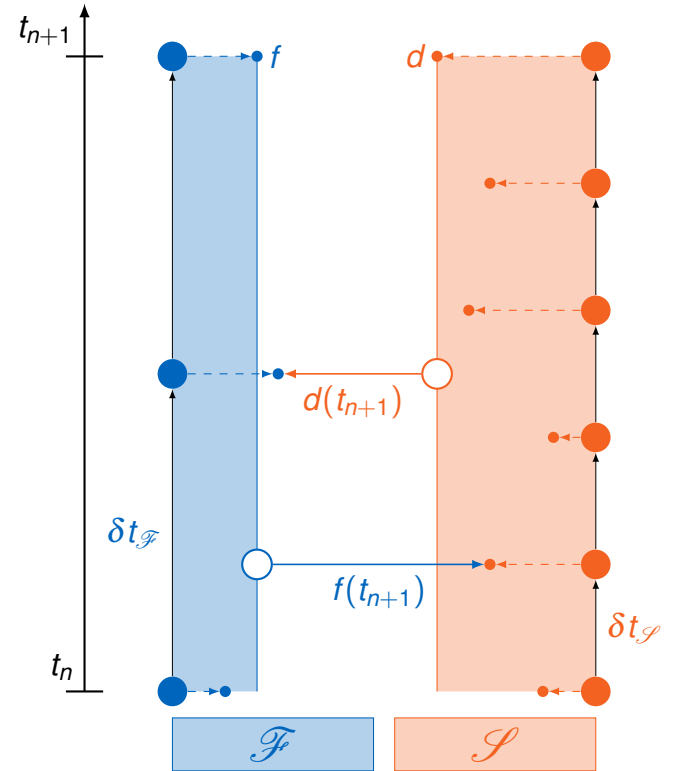
<sup>1</sup><https://github.com/precice/precice/releases/tag/v3.0.0>

# API for implicit Euler

```
1 # example: FSI coupling, perspective of fluid solver  $\mathcal{F}$ :
2 participant = precice.Participant("Fluid", "precice-config.xml")
3
4 # leaving out coupling mesh and data initialization
5
6 participant.initialize()
7
8 while participant.is_coupling_ongoing():
9     # store checkpoint, if needed
10    dt = participant.get_max_time_step_size()
11    displacement = participant.read_data(dt) # Read displacement at  $t_n + \Delta t$ :  $d_{n+1}$ 
12    forces = forces + dt * dfdt(displacements) # Implicit Euler
13    participant.write_data(forces)
14    participant.advance(dt)
15    # read checkpoint, if needed
16
17 participant.finalize()
```

# preCICE v2: single-value coupling

```
<data:vector name="Force" />
<data:vector name="Displ" />
...
<coupling-scheme:serial-implicit>
  <participants first="Fluid" second="Solid" />
  <exchange
    data="Force"
    from="Fluid"
    to="Solid" />
  <exchange
    data="Displ"
    from="Solid"
    to="Fluid" />
  ...
</coupling-scheme:serial-implicit>
```

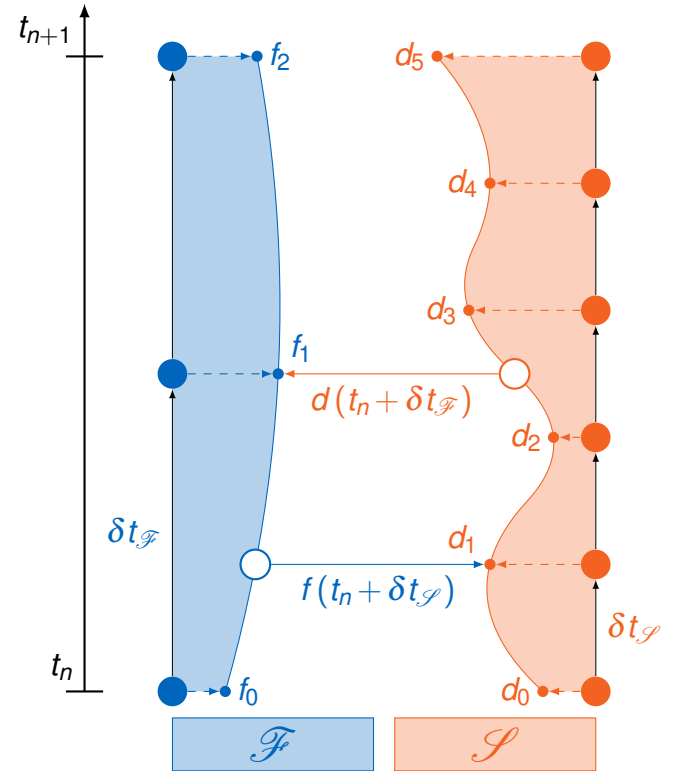


# preCICE v3: waveform iteration

$$\mathcal{F}(\mathcal{S}(f^k)) = \tilde{f}^k$$

upgrade to preCICE v3

$$\mathcal{F}(\mathcal{S}(f(t)^k)) = \tilde{f}(t)^k$$





# RK4 in code (simplified)

```
1 # determine time step size  $\delta t$  for this time step
2 dt = participant.get_max_time_step_size()
3
4 # compute stages  $k_i$  and evaluate waveform at times  $c_i \delta t$ 
5 k1 = A.dot(u) + participant.read_data("Displ", 0.0 * dt)
6 k2 = A.dot(u + k1 * 0.5 * dt) + participant.read_data("Displ", 0.5 * dt)
7 k3 = A.dot(u + k2 * 0.5 * dt) + participant.read_data("Displ", 0.5 * dt)
8 k4 = A.dot(u + k3 * 1.0 * dt) + participant.read_data("Displ", 1.0 * dt)
9 # assemble new solution
10 u_new = u + dt / 6 * (k1 + 2 * k2 + 2 * k3 + k4)
11
12 # do writing
13 force = extract_force(u_new)
14 participant.write_data("Force", force)
15 # end time step of size  $\delta t$ 
16 precice_dt = participant.advance(dt)
```

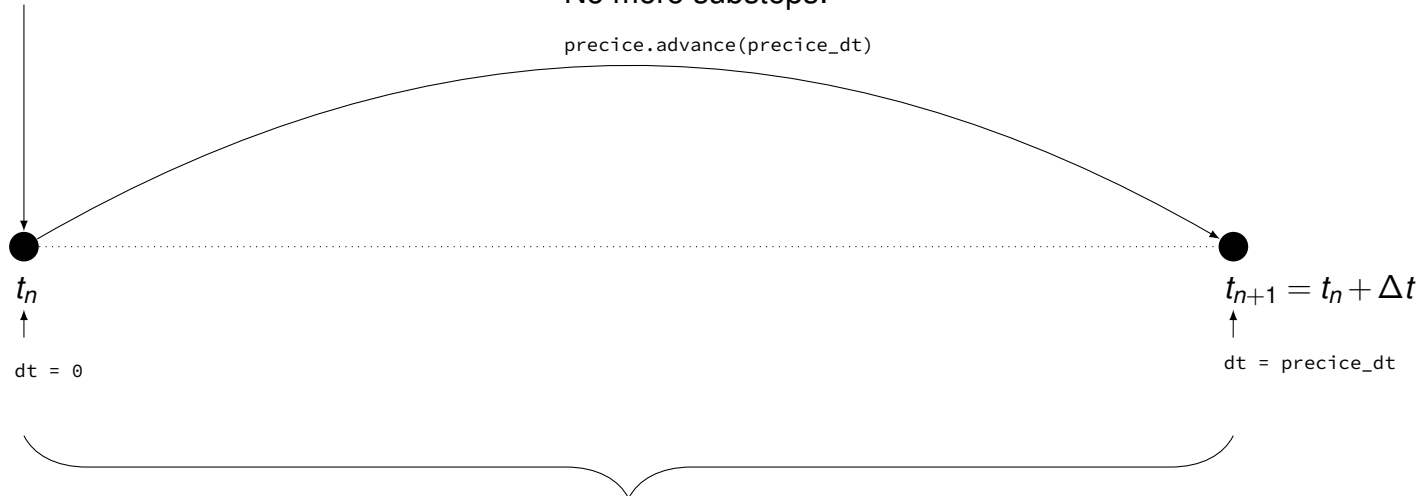
# API for multirate

Read interpolated data from current time  $t_n$ :

```
displacement = precice.read_data(dt)
```

No more substeps:

```
precice.advance(precice_dt)
```



Complete time window size  $\Delta t$

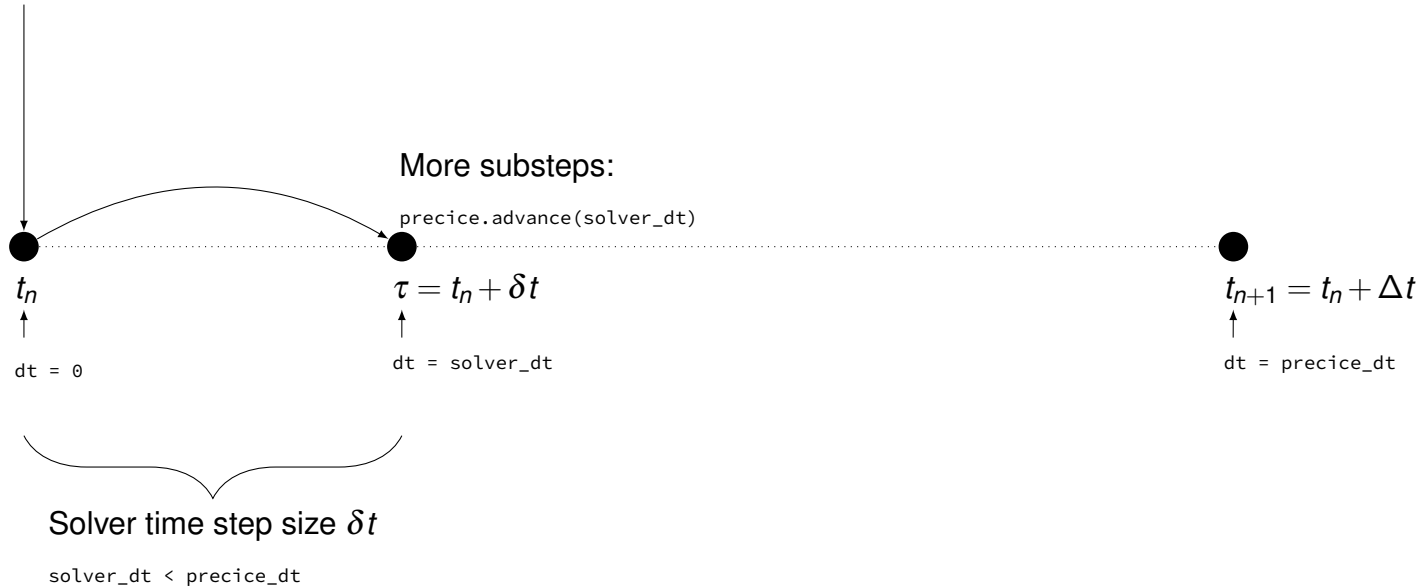
```
precice_dt = precice.get_max_time_step_size()
```



# API for multirate

Read interpolated data from current time  $t_n$ :

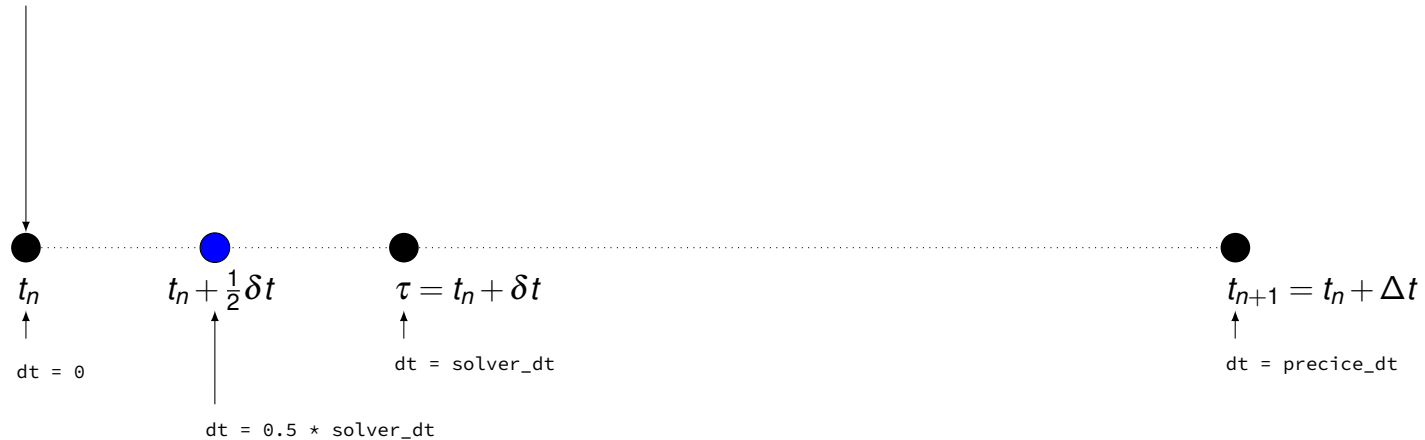
```
displacement = precice.read_data(dt)
```



# API for multirate

Read interpolated data from current time  $t_n$ :

```
displacement = precice.read_data(dt)
```

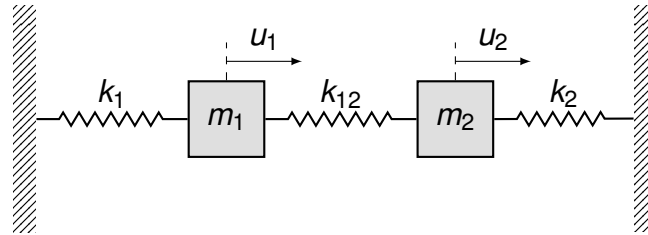


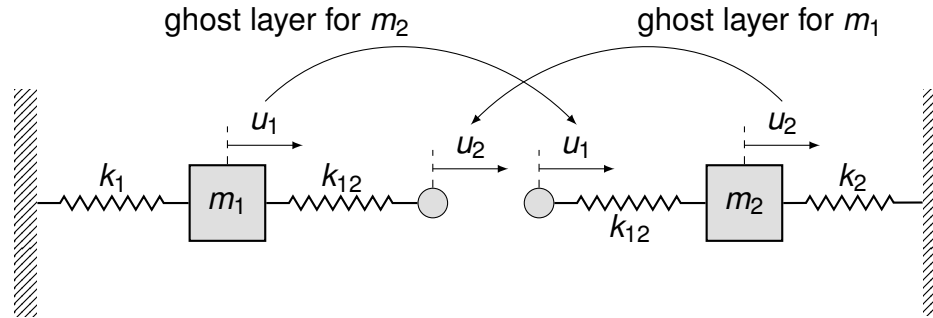
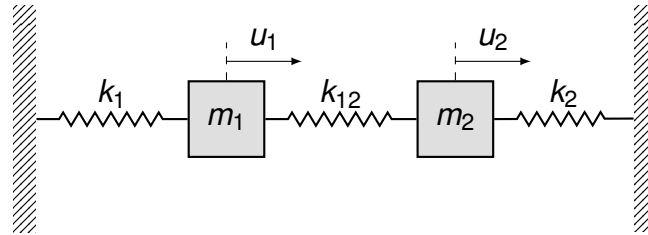
# API with RK4 and multirate

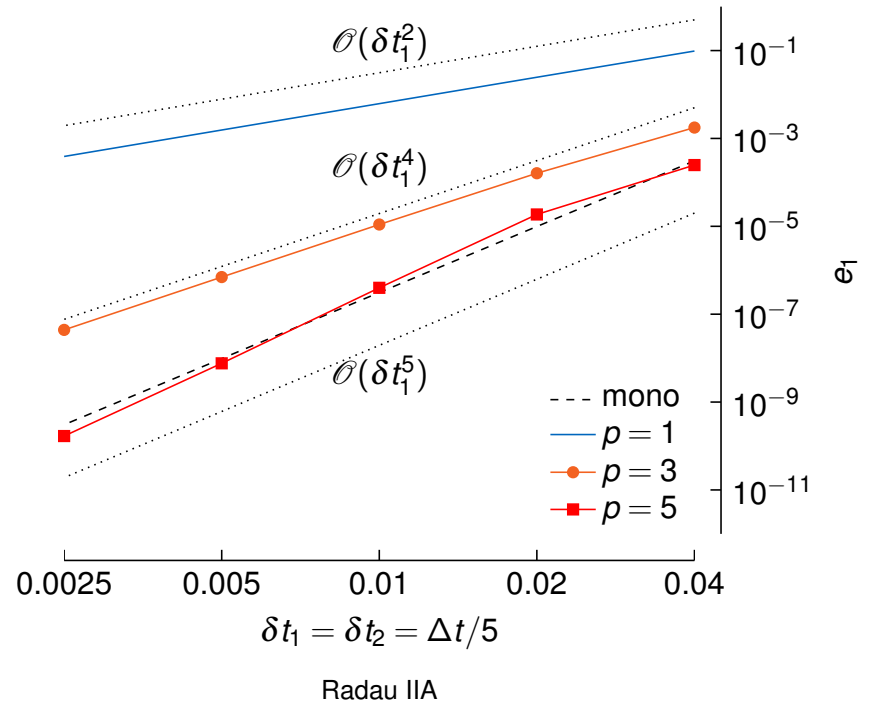
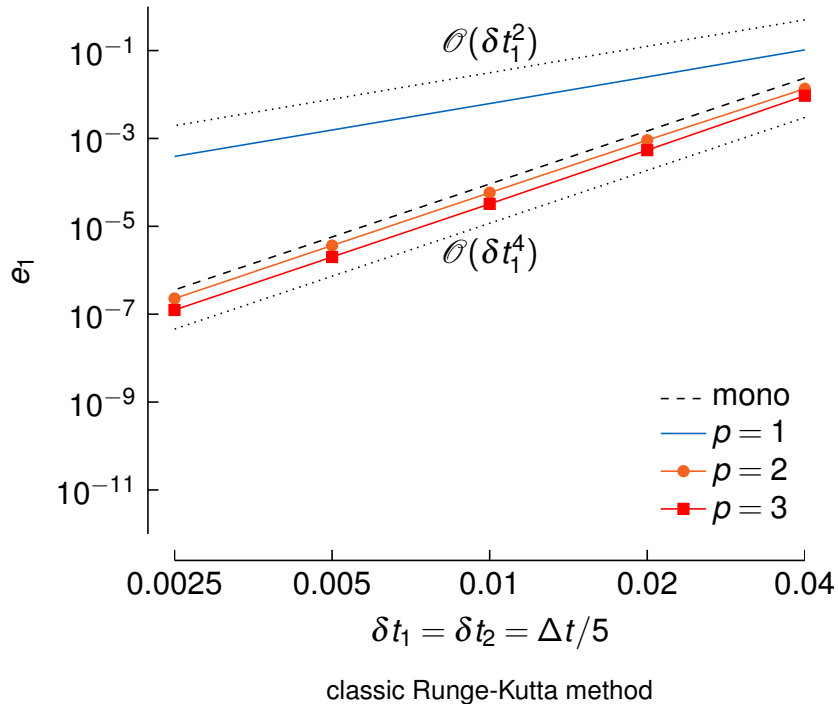
```

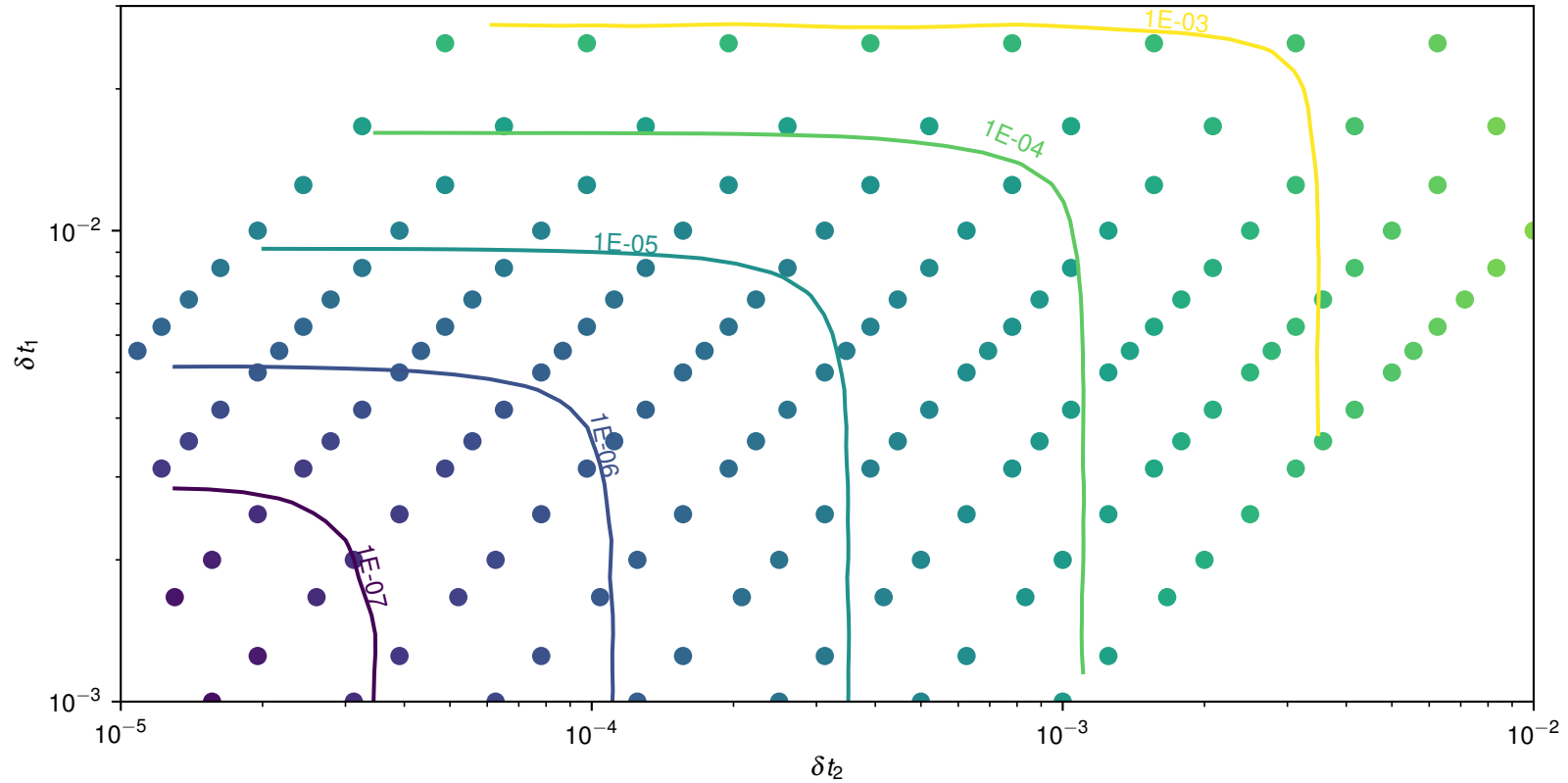
1 # example: FSI coupling, perspective of fluid solver  $\mathcal{F}$ :
2 # ...
3 participant.initialize()
4
5 while participant.is_coupling_ongoing():
6     # store checkpoint, if needed
7     precice_dt = participant.get_max_time_step_size() # until end of window
8     solver_dt = time_stepper.get_max_dt() # stability, adaptivity
9     dt = np.min([precice_dt, solver_dt]) # actual time step size  $\delta t$ 
10    # time_stepper represents s-stage RK scheme
11    ts = time_stepper.rhs_eval_points(dt) #  $t_n + c_1, t_n + c_2, \dots, t_n + c_s$ 
12    displacements = [participant.read_data(t) for t in ts] #  $d(t_n + c_1), d(t_n + c_2), \dots, d(t_n + c_s)$ 
13    forces = time_stepper.do_step(displacements, dt) #  $f_{n+1} = f_n + \delta t \sum_{i=1}^s b_i k_i$ 
14    participant.write_data(forces)
15    participant.advance(dt)
16    # read checkpoint, if needed
17
18 participant.finalize()

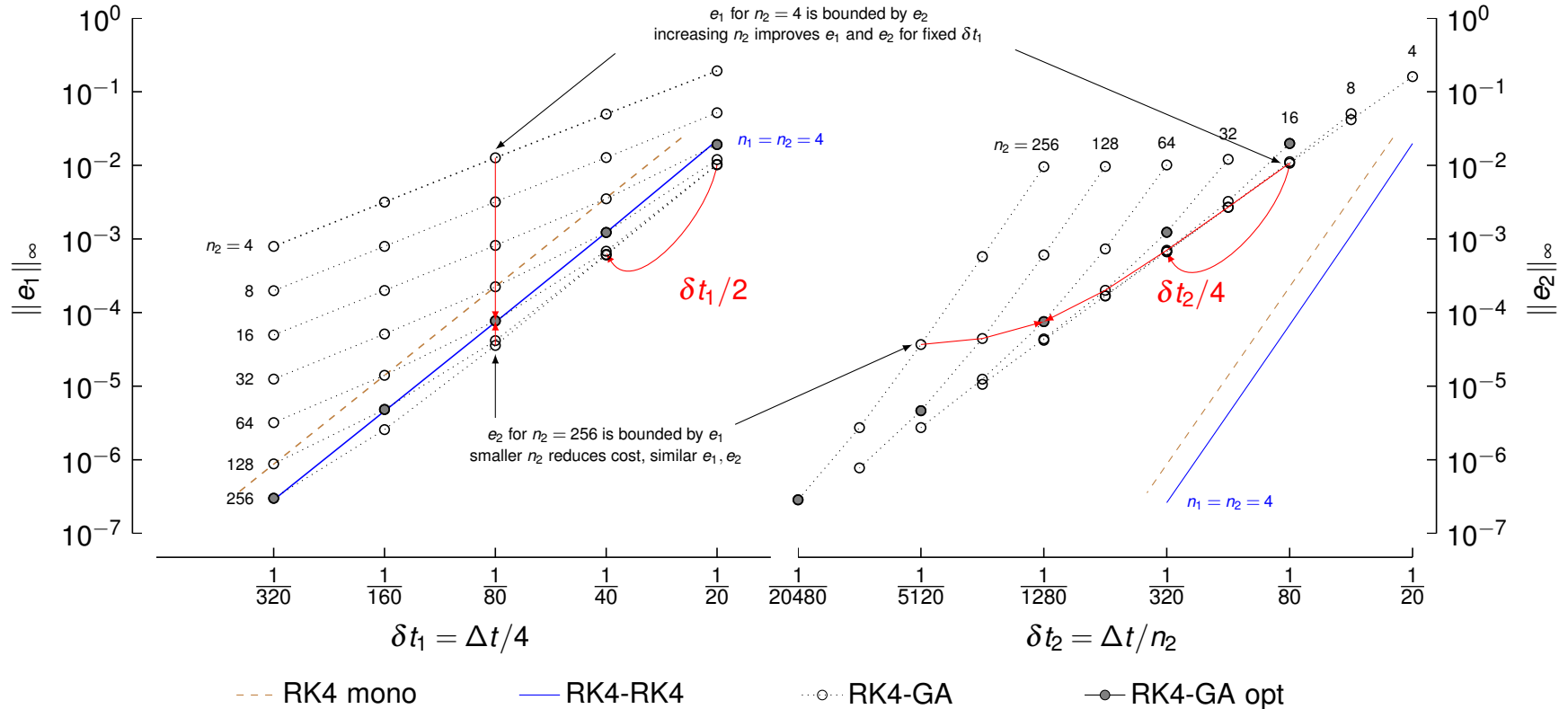
```



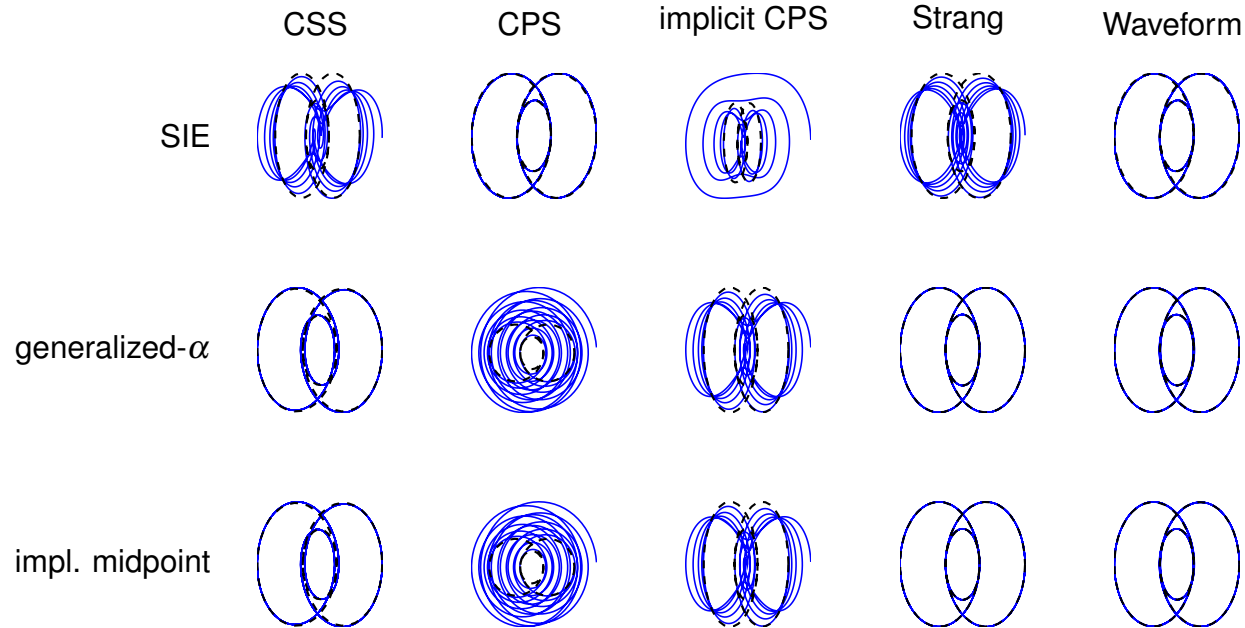




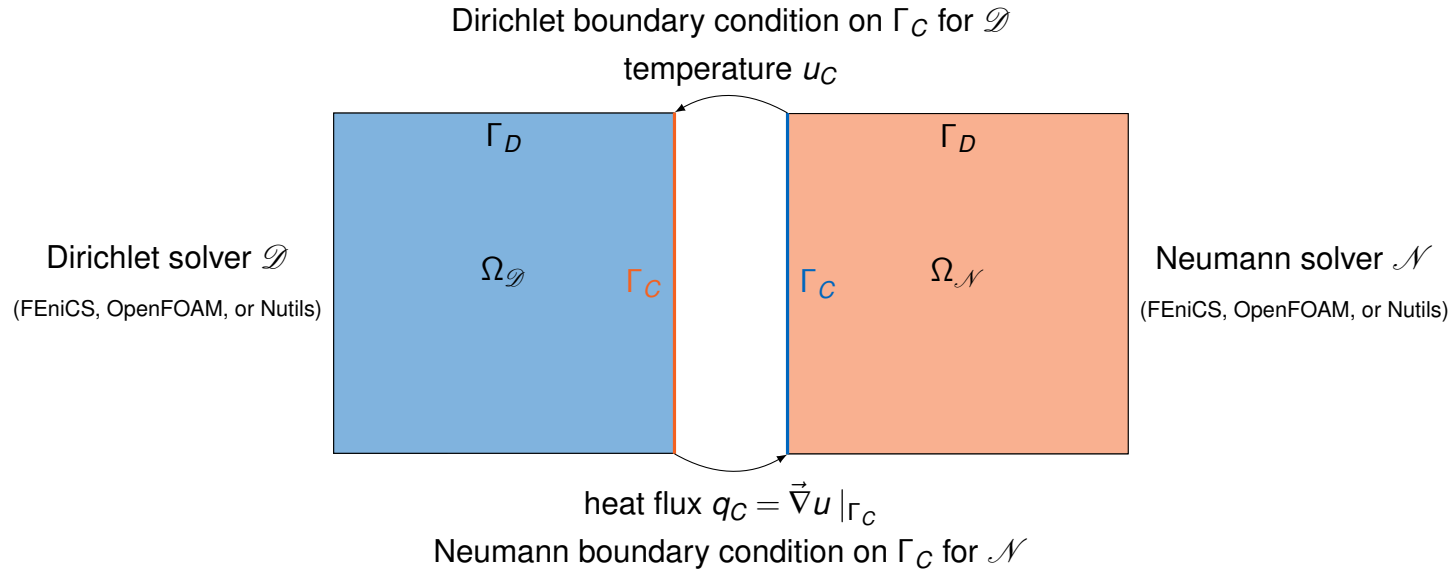


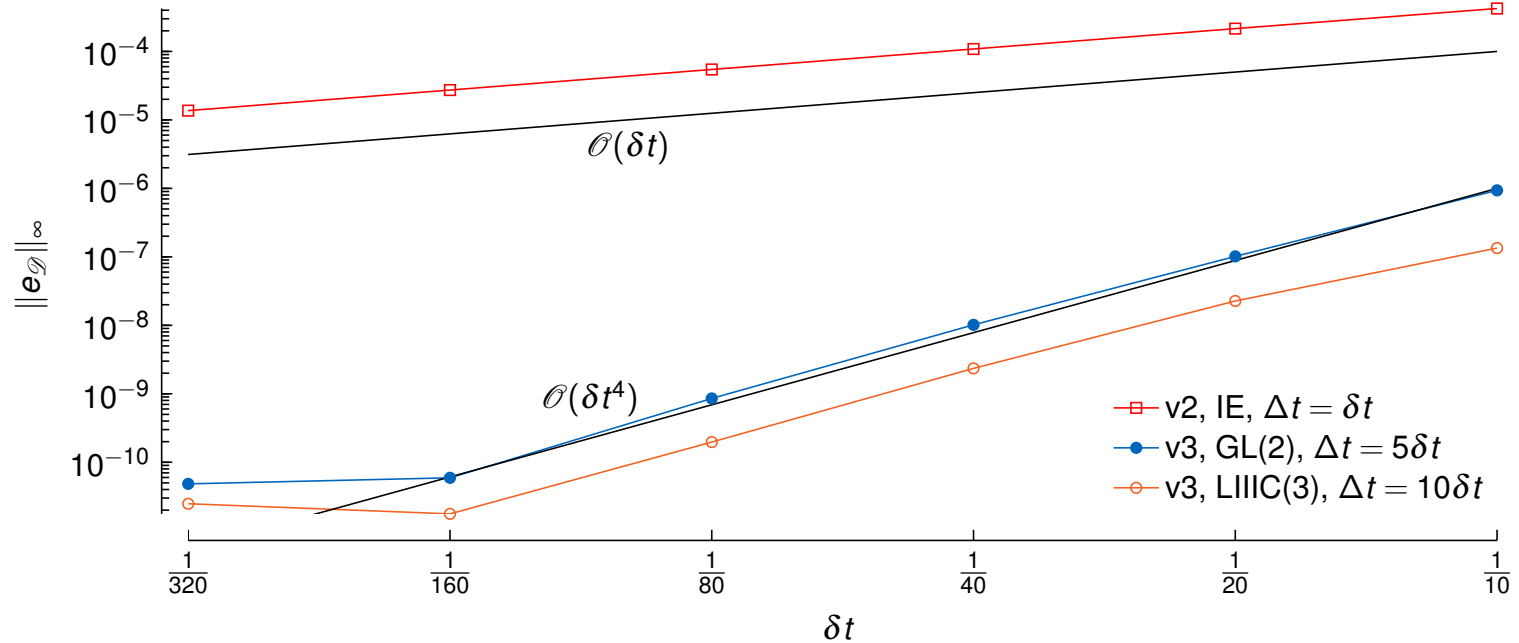






*A Simple Test Case for Convergence Order in Time and Energy Conservation of Black-Box Coupling Schemes. 2022.*





Bachelor's thesis by Niklas Vinnitchenko *Evaluation of Higher-Order Coupling Schemes with FEniCS-preCICE*

# tutorials/partitioned-heat-conduction



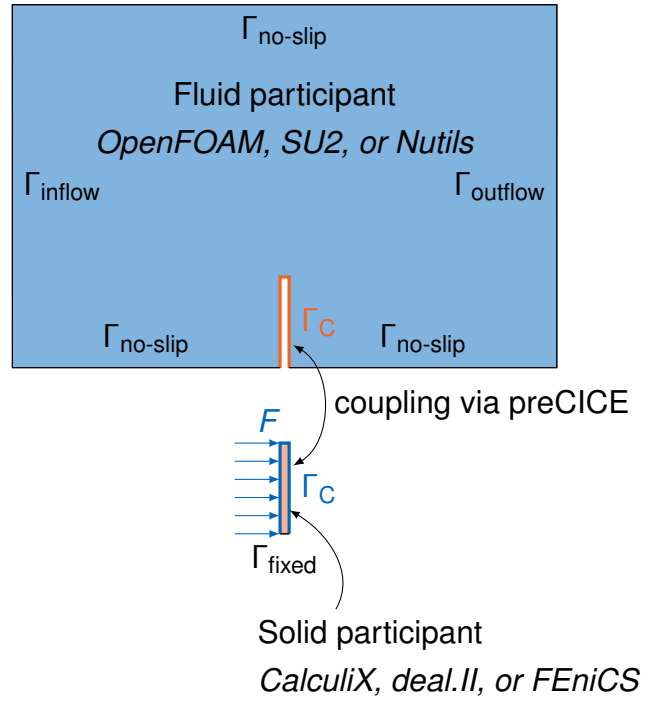
## Tooling: Higher-order time stepping

- FEniCS + pySDC
- [github.com/firedrakeproject/Irksome](https://github.com/firedrakeproject/Irksome)<sup>1</sup>  
 $\text{inner}(\text{Dt}(u), v) * dx + \text{inner}(\text{grad}(u), \text{grad}(v)) * dx - \text{inner}(\text{rhs}, v) * dx$
- Applied general approach to FEniCS:
  - B.Sc. thesis Nikola Wullenweber monolithic higher-order
  - B.Sc. thesis Niklas Vinnitchenko partitioned higher-order
  - Need derivatives of B-splines!

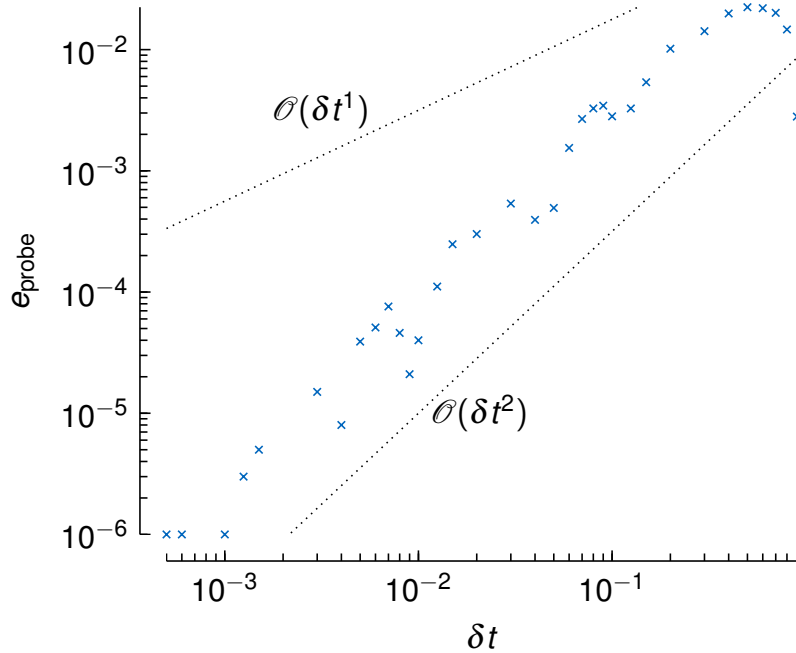
---

<sup>1</sup>Farrell, Patrick E., Robert C. Kirby, and Jorge Marchena-Menendez. *Irksome: Automating Runge–Kutta time-stepping for finite element methods*

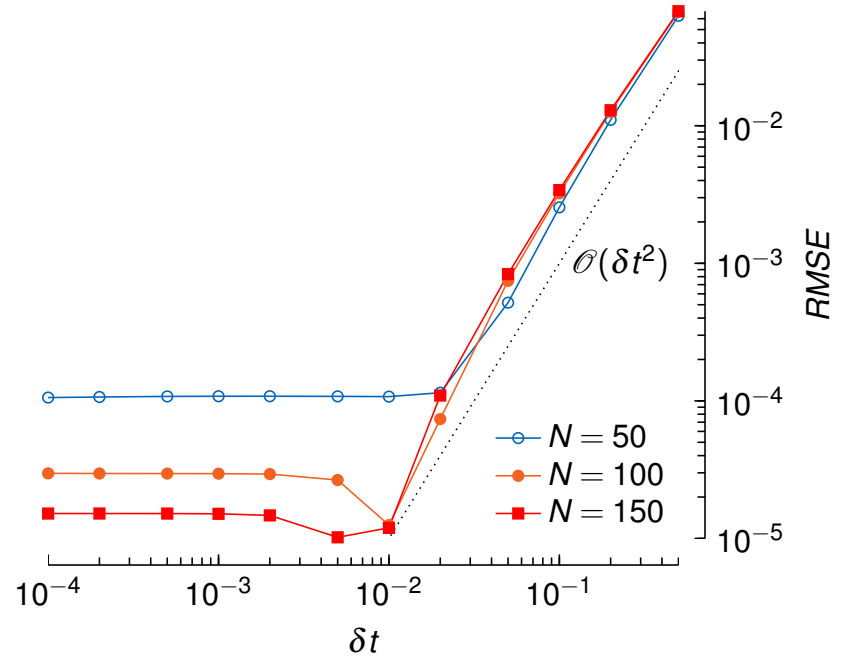
# tutorials/perpendicular-flap (without coupling)



# tutorials/perpendicular-flap (without coupling)



CalculiX beam (Probe at beam tip, compare to  $\delta t = 10^{-5}$ )



OpenFOAM Taylor-Green vortex

Guided research project by Marc Amorós Trepát *Review of higher-order time stepping schemes in open-source solvers*

# Conclusion

## State of development

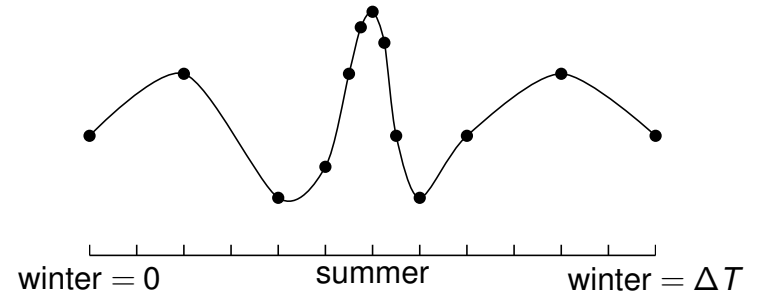
- Multirate time-stepping + black-box + usable API
- Time interpolation new in preCICE v3
- First tutorials use time interpolation

## Where to be careful:

- Requirement: High order interpolation needs subcycling
- Restriction: Only simple acceleration with substeps, <https://github.com/precice/precice/pull/1834>

## Many questions:

- Synchronization, subcycling, and performance?
- Fluid-structure interaction case?
- Adaptivity (inside window / across windows)



# Community



preCICE Workshop 2023@Munich

## Stay in touch?

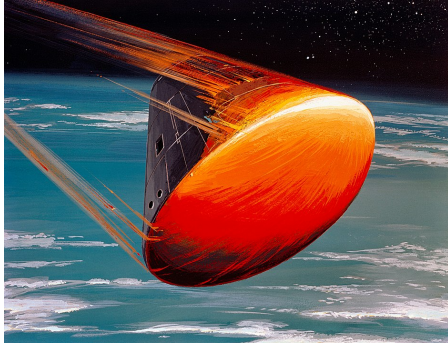
- [precice.org/community](https://precice.org/community)
- [precice.discourse.group](https://precice.discourse.group)

## Conferences

- WCCM + preCICE course  
July 2024@Vancouver
- preCICE Workshop  
Sept 2024@Stuttgart
- COUPLED  
2025@Sicily



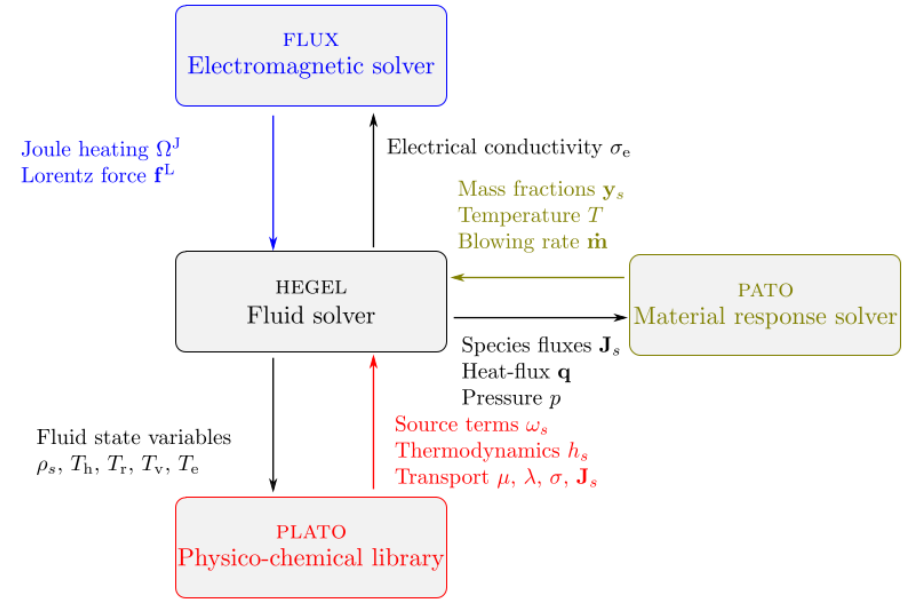
# Rocket science



By North American Rockwell, Public Domain,  
<https://commons.wikimedia.org/w/index.php?curid=2466251>

## Partitioned solver for ICP<sup>1</sup> wind tunnels

- $\Delta t_{\text{FLUX}} \approx 10 \Delta t_{\text{HEGEL}} \approx 1000 \Delta t_{\text{PATO}}$
- HEGEL uses 3rd order RK
- PATO uses  $\Delta t_{\text{PATO}} = 10^{-4} \text{s} = \Delta T_{\text{preCICE}} / 100$
- HEGEL uses  $\Delta t_{\text{HEGEL}} = 10^{-2} \text{s} = \Delta T_{\text{preCICE}}$
- FLUX uses  $\Delta t_{\text{FLUX}} = 10^{-1} \text{s} = 10 \Delta T_{\text{preCICE}}$  (???)



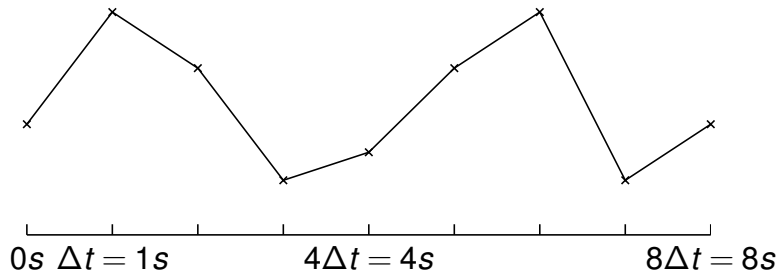
from Alessandro Munafò, et al. *A Multi-Physics Modeling Framework for Inductively Coupled Plasma Wind Tunnels*. 2022.  
<https://doi.org/10.2514/6.2022-1011>

<sup>1</sup>Inductively Coupled Plasma

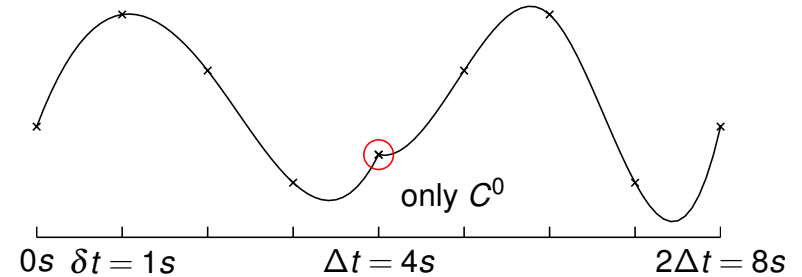
# Subcycling

- Time window size  $\Delta t \geq$  time step size  $\delta t_1$  and  $\delta t_2$ .
- Do  $n$  time steps in window:  $\Delta t = n_1 \delta t_1 = n_2 \delta t_2$
- Allows to create BSpline of degree  $n - 1$ . (Goal reached: Something better than linear interpolation)
- Restriction: Only use data of current window!
- Larger window + subcycling has impact on number of QN iterations<sup>1</sup>

Without subcycling



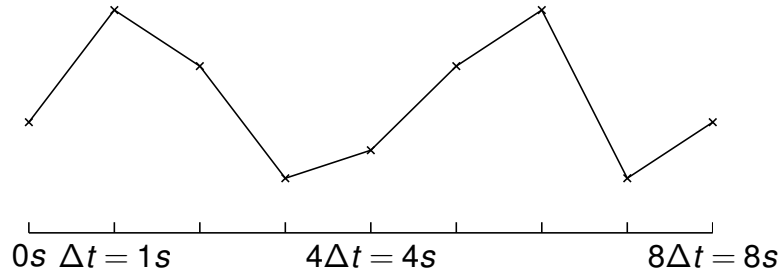
With subcycling (third order BSpline)



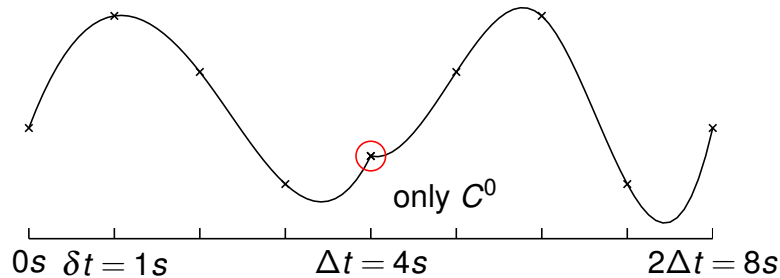
<sup>1</sup>Rüth, B, Uekermann, B, Mehl, M, Birken, P, Monge, A, Bungartz, H-J. Quasi-Newton waveform iteration for partitioned surface-coupled multiphysics applications. Int J Numer Methods Eng. 2021; 122: 5236– 5257. <https://doi.org/10.1002/nme.6443>

# QN iterations

Without subcycling



With subcycling (third order BSpline)



rQN-WI	$\Delta t$	0.5	0.1
WI(1, 1; 1)		7.85	5.45
WI(5, 5; 1)		10.95	7.48

**rQN-WI** means we only use the data at the end of the window for Quasi-Newton. Different example case, but similar implementation. More possibilities shown in<sup>1</sup>.

<sup>1</sup>Rüth, B, Uekermann, B, Mehl, M, Birken, P, Monge, A, Bungartz, H-J. Quasi-Newton waveform iteration for partitioned surface-coupled multiphysics applications. Int J Numer Methods Eng. 2021; 122: 5236– 5257. <https://doi.org/10.1002/nme.6443>

$$\Delta t \gg \delta t$$

