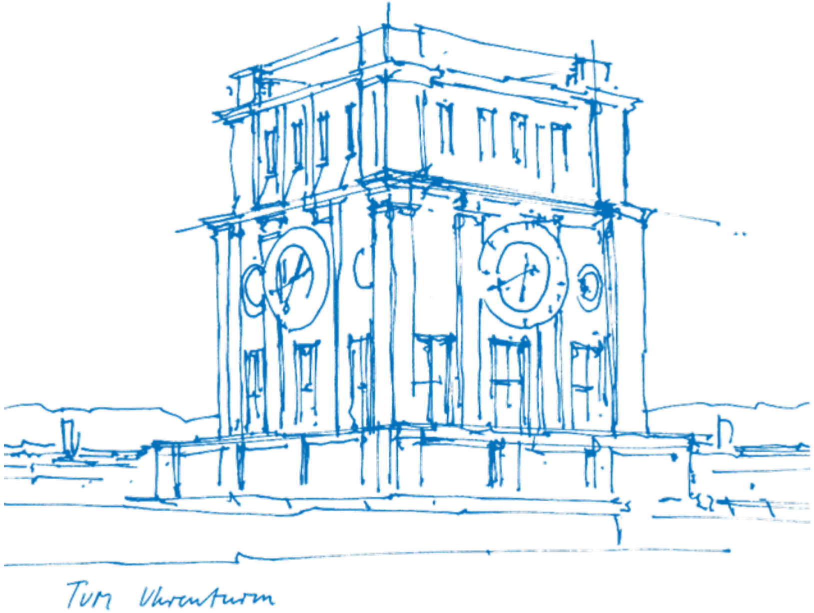
High-order and multi-rate time stepping with preCICE

Benjamin Rüth

Technical University of Munich, Department of Informatics



preCICE Workshop 2020
Garching, Germany
February 5, 2020

new preCICE release v2.0 is ready to use - Inbox - Mozilla Thunderbird

Inbox | new preCICE release v2 ×

Get Messages | Write | Chat | Address Book | Tag | Quick Filter | Search <Ctrl+K>

Reply | Reply List | Forward | Archive | Junk | Delete | More

From Benjamin Uekermann <benjamin.uekermann@gmail.com>

Subject **new preCICE release v2.0 is ready to use** 09:32

To preCICE <precice@mailman.informatik.uni-stuttgart.de>

The most important changes:

- You can now (finally) use the same preCICE configuration file for serial and parallel runs.
- The configuration reference can now also be generated in markdown. A recent version is always in the wiki.
- We cleaned up the duplicated meaning of "timestep". preCICE now uses "time windows" and the participants do their "timesteps".
- Mesh handling is much faster 🚀.
- We moved the Python and (**new**) Matlab bindings as well as the Fortran Module (formerly known as f2003 bindings) to separate repositories. Btw, the Python bindings are now really pythonic and you can get them through PyPI.
- For the first time, two-level initialization is available, allowing for fast initialization of very large cases 🚀🚀. The feature is, however, still in beta testing and switched off by default. We will have a presentation at the workshop (and afterwards online) about the new initialization concept.
- We restructured the repository a bit: developer tools are now in *tools*, user tools in *extras*, native bindings in *extras/bindings*, and solver dummies in *examples*. These examples are now also shipped with our binary packages, and you can use them to test your installation.

Write: Re: new preCICE release v2.0 is ready to use - Thunderbird

File Edit View Insert Format Options Tools Help

Send | Spelling | Security | Save | Attach

From: Benjamin Rüth <rueth@in.tum.de> rueth@in.tum.de

To: Benjamin Uekermann <benjamin.uekermann@gmail.com>

Subject: Re: new preCICE release v2.0 is ready to use

Paragraph | Variable Width

Dear Benjamin,

I do not really know what a time window is supposed to be and why it is better than a timestep.

Is it really necessary to torture the whole community???

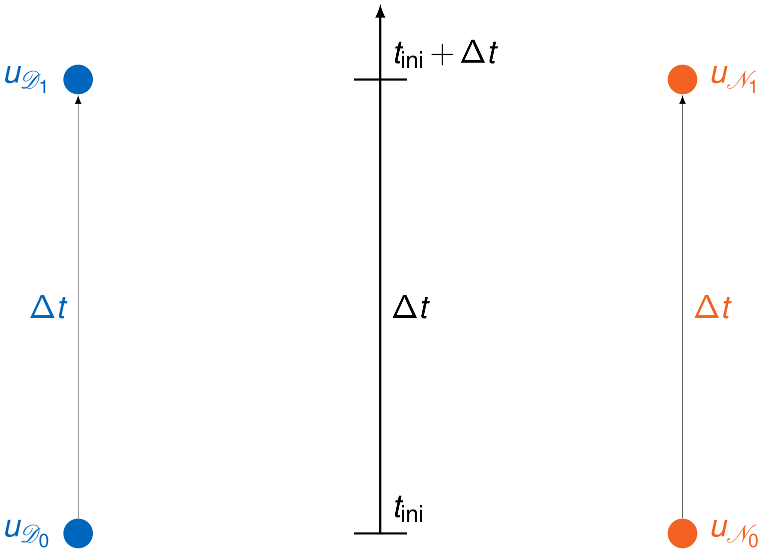
Best regards,

a preCICE user

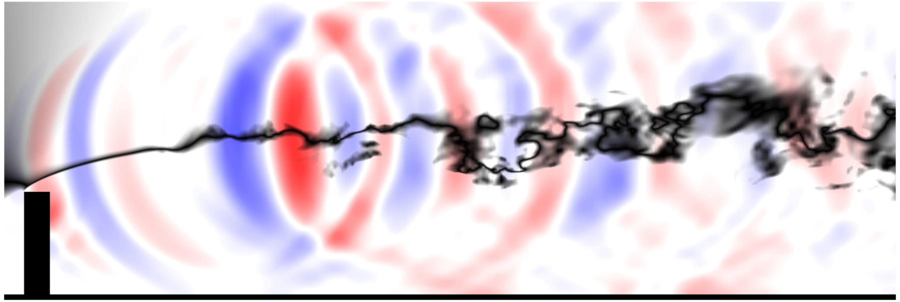
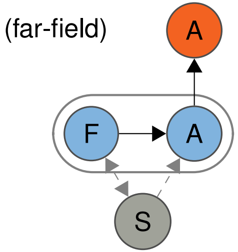
On 14.02.20 09:32, Benjamin Uekermann wrote:

> Dear preCICE Community,
>
> You might have seen it on GitHub already: we have a fresh new release, preCICE v2.0 🎉 and since yesterday, all adapters, bindings, and tutorials are compatible.
>
> Breaking news: we have breaking changes ☺. We decided to move to v2.0 to clean up some

How time stepping usually works



ExaFSA setup

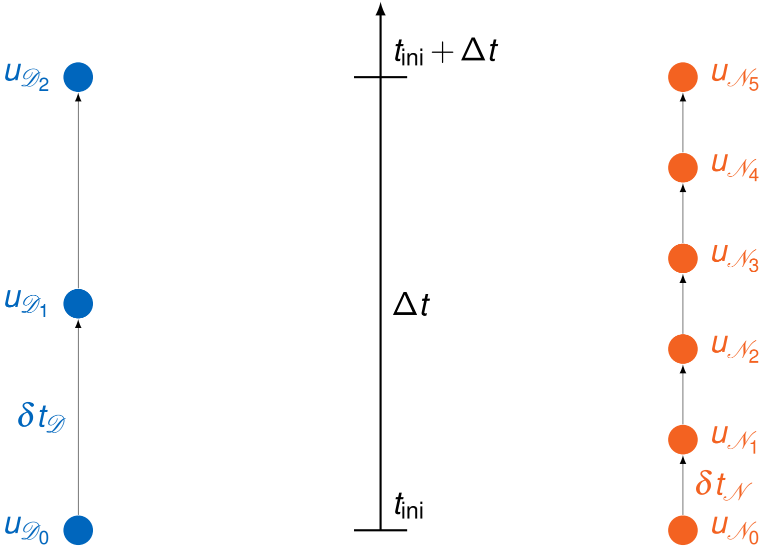
Fluid-acoustics simulation and partitioned setup[1].

physics	timescale	solver
(A)	small	Ateles
(A)	small	FASTEST
(F)	medium	FASTEST
(S)	large	FEAP

- Quasi-Newton
- Black-box
- High order time-stepping

[1] Reimann, T., et al. (2017). Aspects of FSI with aeroacoustics in turbulent flow. In 7th GACM Colloquium on Computational Mechanics.

Let's use multirate/subcycling!



Let's use multirate/subcycling!

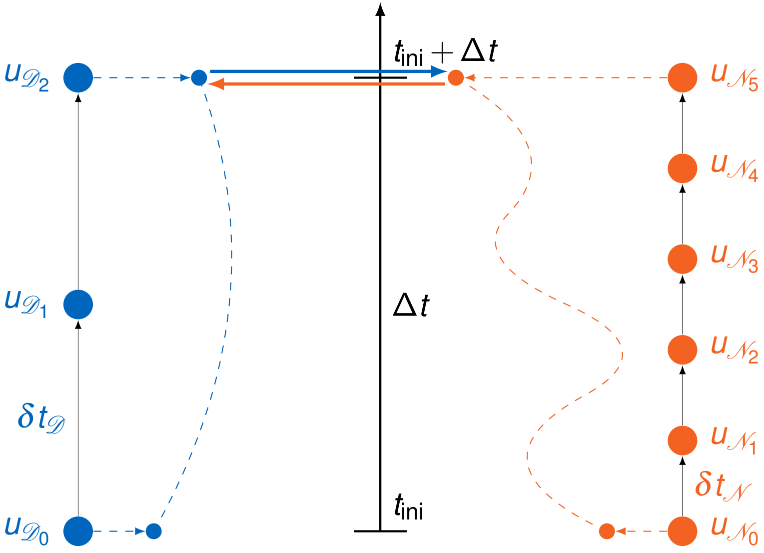
```
double solver_dt = 0.1; // solver timestep size
double precice_dt; // maximum precice timestep size
double t = 0;   // time

precice_dt = precice.initialize();  // e.g. 0.5

while (precice.isCouplingOngoing()){

    ...  // reading

    dt = min(precice_dt, solver_dt);  // always 0.1
    solver.doTimestep(dt);
    t += dt;  //                              0.1; 0.2; 0.3; 0.4; 0.5; 0.6, 0.7 ...
    precice_dt = precice.advance(dt);  // 0.4; 0.3; 0.2; 0.1; 0.5; 0.4, 0.3 ...

    ... // writing

}
```

Let’s use multirate/subcycling!

```
double solver_dt = 0.1; // solver timestep size
double precice_dt; // maximum precice timestep size
double t = 0;   // time

precice_dt = precice.initialize();  // e.g. 0.5

while (precice.isCouplingOngoing()){

    ...  // reading + save checkpoint

    dt = min(precice_dt, solver_dt);  // always 0.1
    solver.doTimestep(dt);
    t += dt;                           // 0.1; 0.2; 0.3; 0.4; 0.5; 0.1, 0.2 ...
    precice_dt = precice.advance(dt);  // 0.4; 0.3; 0.2; 0.1; 0.5; 0.4, 0.3 ...

    ... // writing + restore checkpoint

}
```

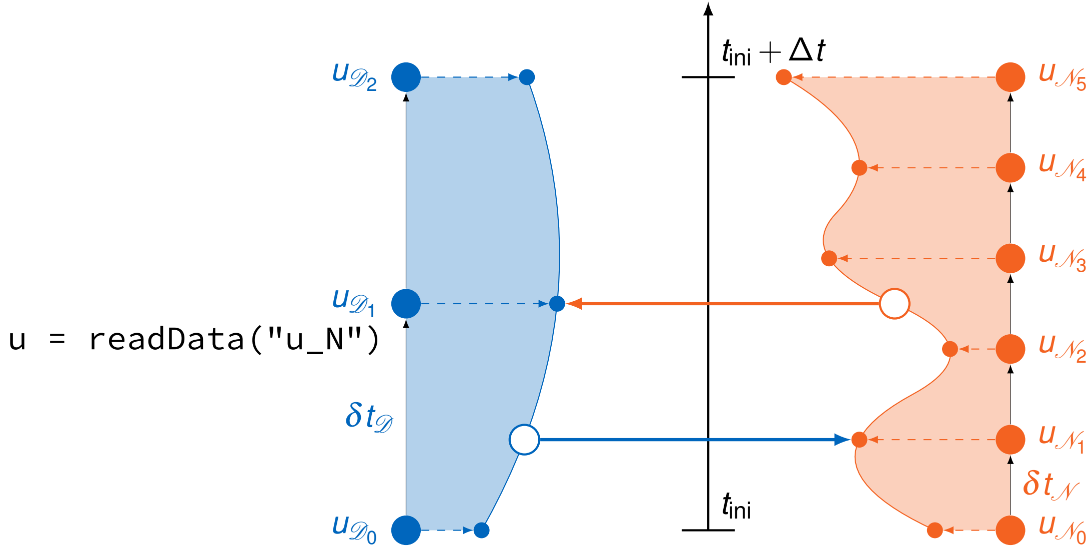
Let's use multirate/subcycling!

BCs are constant over window Δt



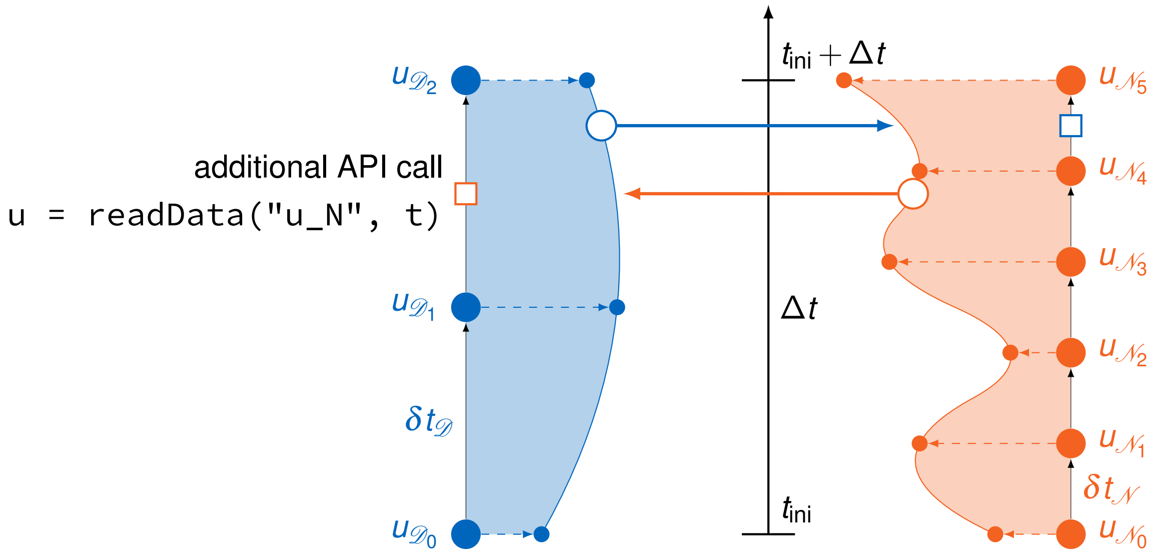
Let’s use multirate/subcycling!

BCs are constant over timestep δt

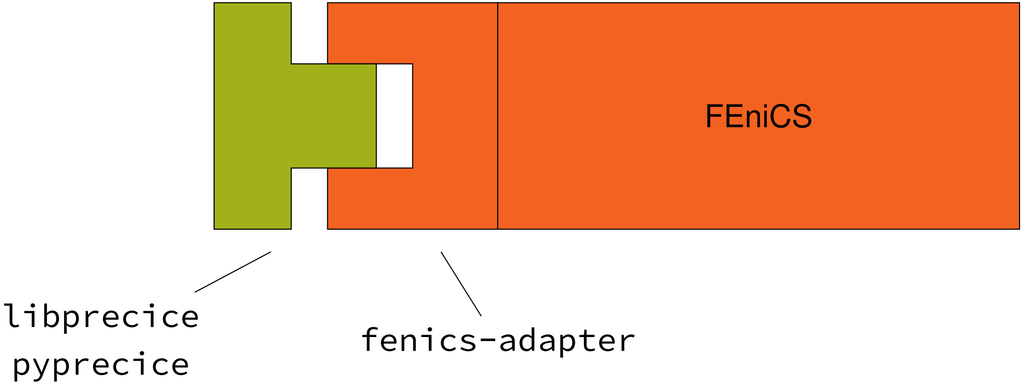


Let’s use multirate/subcycling!

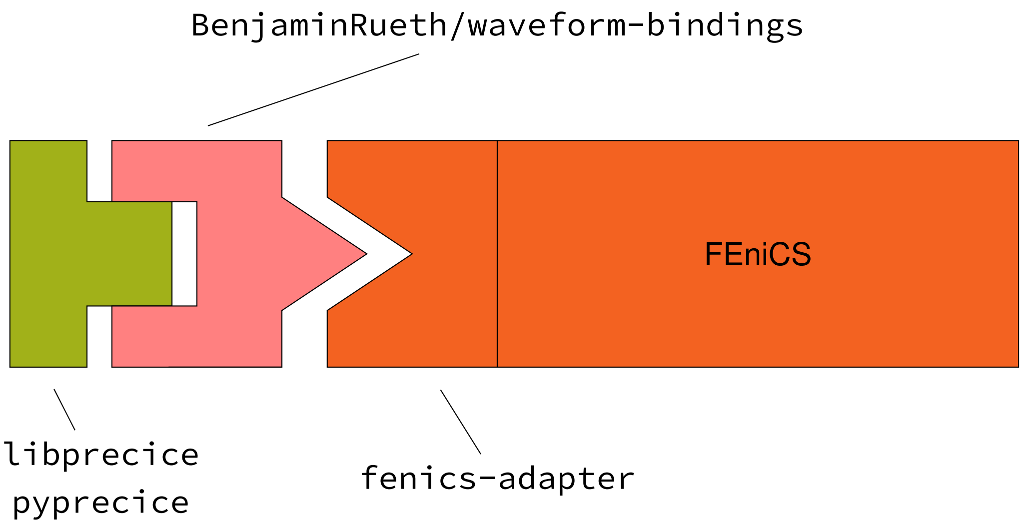
BCs are interpolated over time t



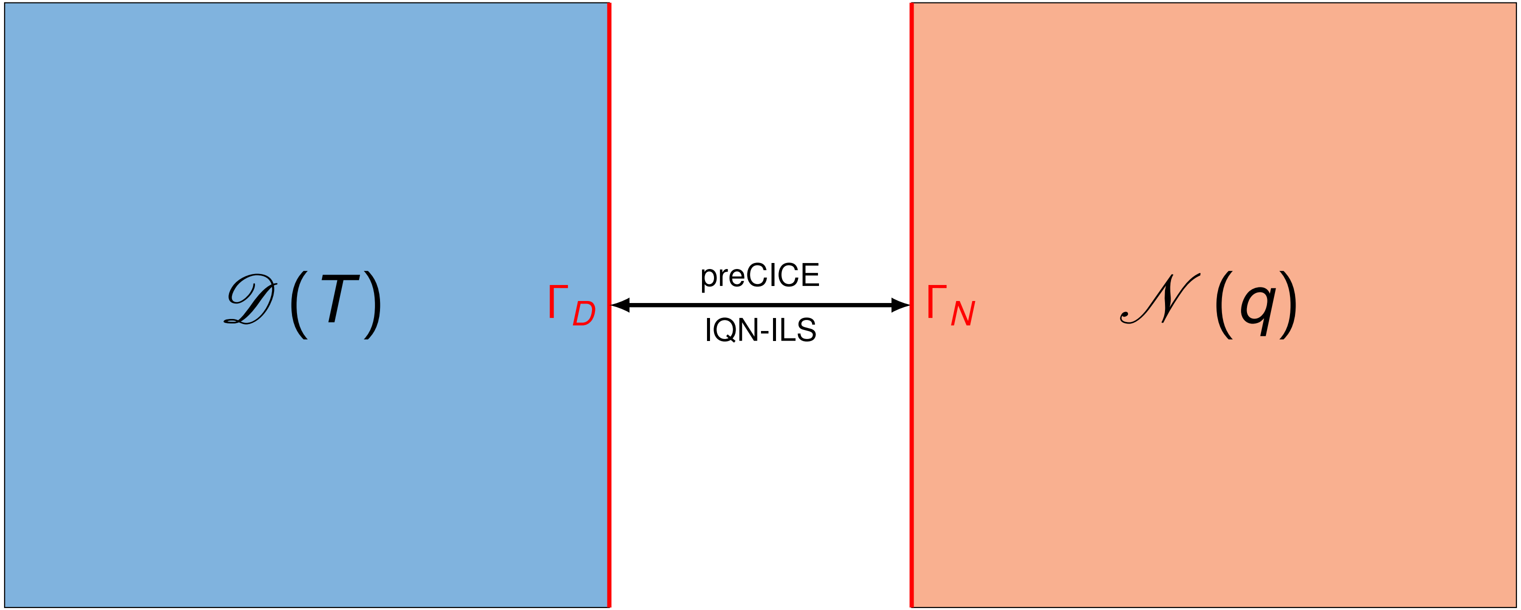
Prototype implementation



Prototype implementation



Partitioned Heat Equation



Partitioned Heat Equation

```xml
<participant name="D">
  <write-data name="q1"/>
  <write-data name="q2"/>
  <read-data  name="T1"/>
  <read-data  name="T2"/>
  <read-data  name="T3"/>
  <read-data  name="T4"/>
  <read-data  name="T5"/>
</participant>
```

preCICE

IQN-ILS

```xml
<participant name="N">
  <write-data name="T1"/>
  <write-data name="T2"/>
  <write-data name="T3"/>
  <write-data name="T4"/>
  <write-data name="T5"/>
  <read-data  name="q1"/>
  <read-data  name="q2"/>
</participant>
```

Partitioned Heat Equation

Check QN Performance

- different multirate setups **WI(**$n_{\mathscr{D}}, n_{\mathscr{N}}$)
- QN-WI feeds all samples into Quasi Newton
- interpolate BCs over time
- only linear interpolation & implicit Euler
- compute for $T = 10$

QN-WI | Δt	5.0	0.5	0.1
WI(1, 1)	10.50	7.85	5.45
WI(1, 3)	11.50	8.85	6.60
WI(1, 5)	11.50	8.75	6.77
WI(3, 1)	10.50	8.10	5.43
WI(3, 3)	12.00	9.30	6.36
WI(3, 5)	12.00	9.85	6.89
WI(5, 1)	10.50	8.15	5.43
WI(5, 3)	11.50	9.85	6.82
WI(5, 5)	12.00	9.45	6.41

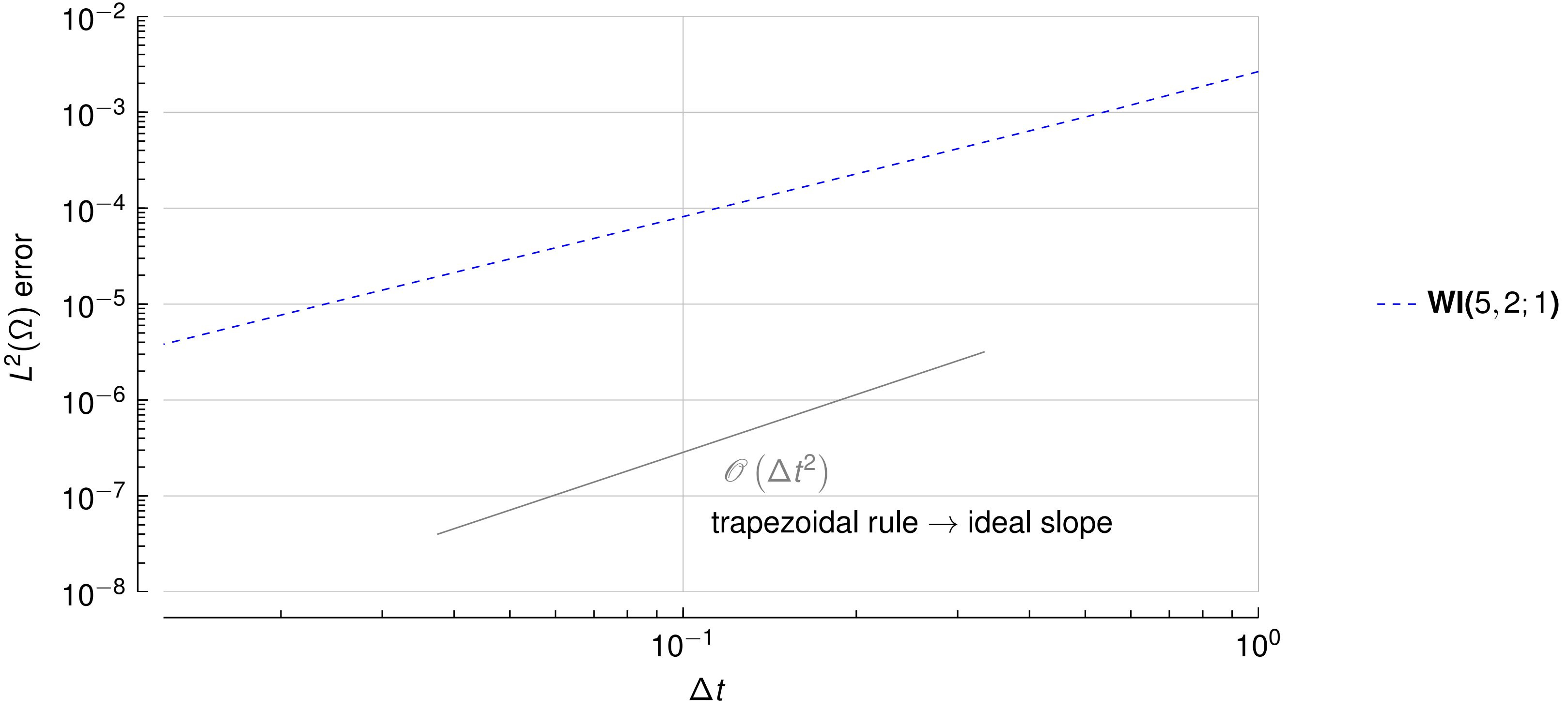
Partitioned Heat Equation

Check QN Performance

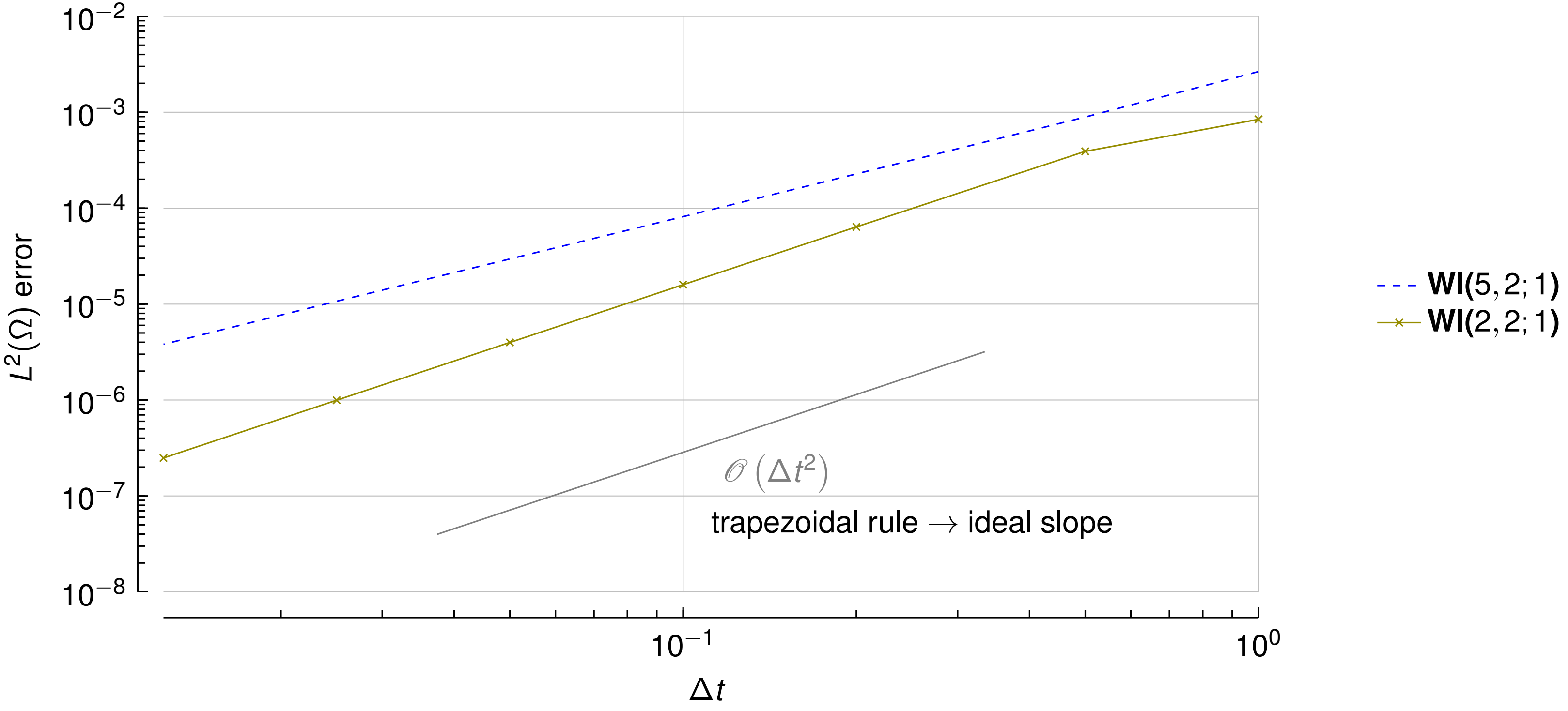
- different multirate setups **WI**($n_{\mathscr{D}}, n_{\mathscr{N}}$)
- QN-WI feeds all samples into Quasi Newton
- interpolate BCs over time
- only linear interpolation & implicit Euler
- compute for $T = 10$

QN-WI | Δt	5.0	0.5	0.1
WI$(1,1)$	10.50	7.85	$\boxed{5.45}$
WI$(1,3)$	11.50	8.85	6.60
WI$(1,5)$	11.50	8.75	6.77
WI$(3,1)$	10.50	8.10	5.43
WI$(3,3)$	12.00	9.30	6.36
WI$(3,5)$	12.00	9.85	6.89
WI$(5,1)$	10.50	8.15	5.43
WI$(5,3)$	11.50	9.85	6.82
WI$(5,5)$	12.00	$\boxed{9.45}$	6.41

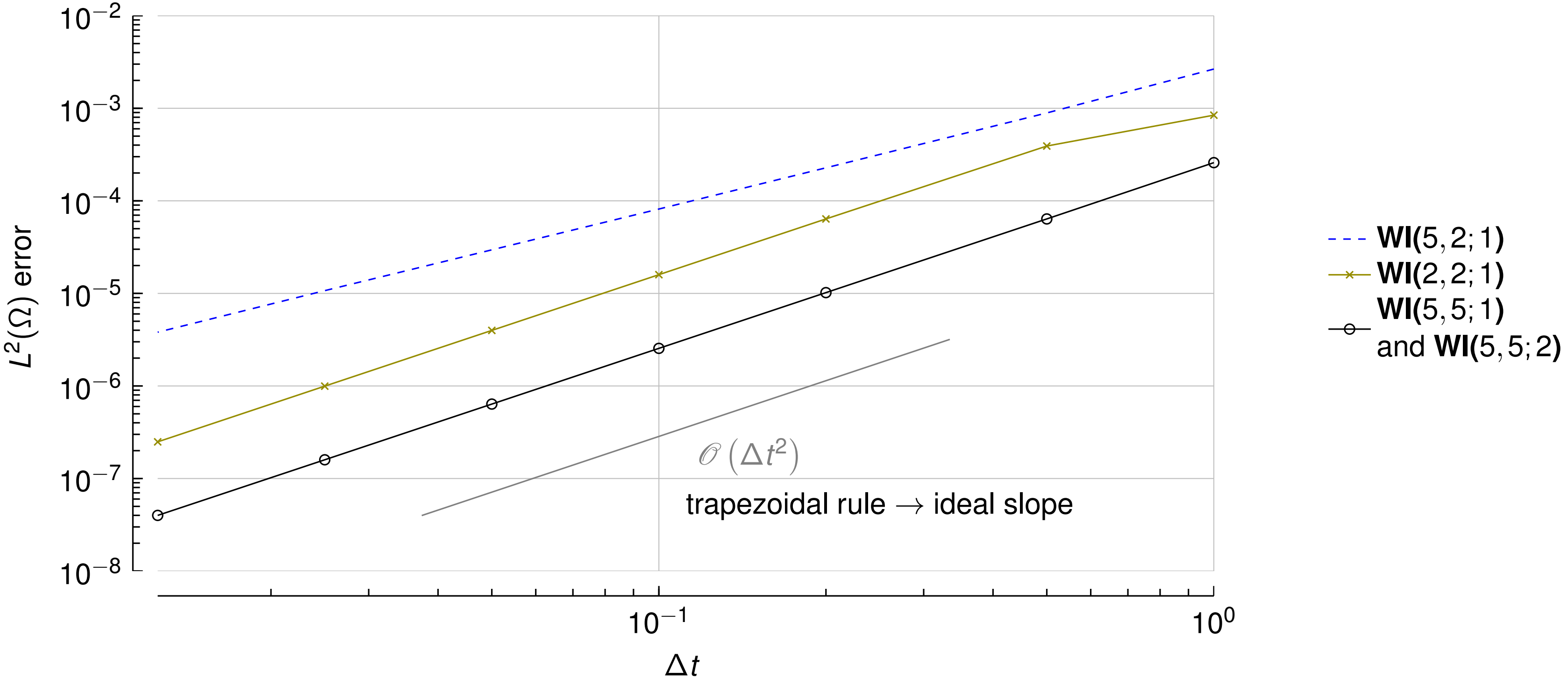
Partitioned Heat Equation



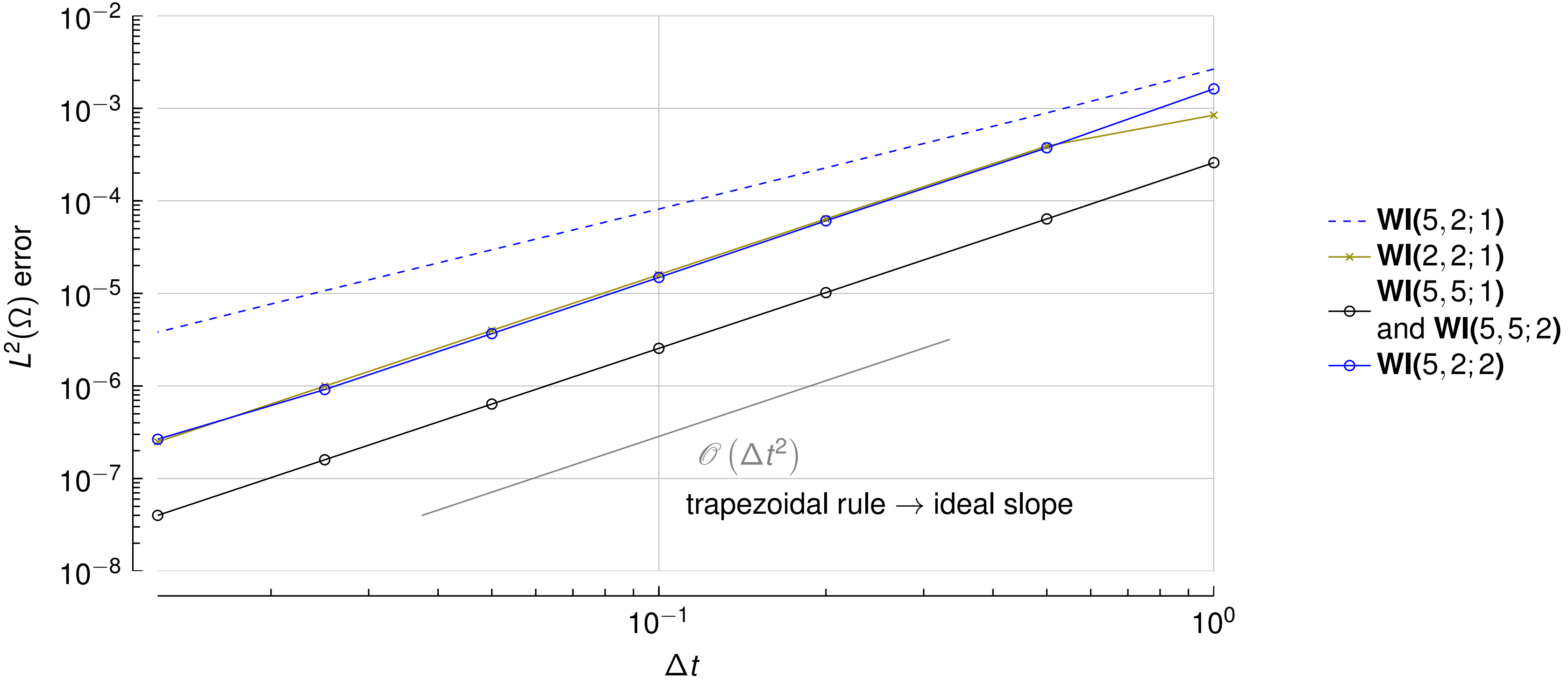
Partitioned Heat Equation



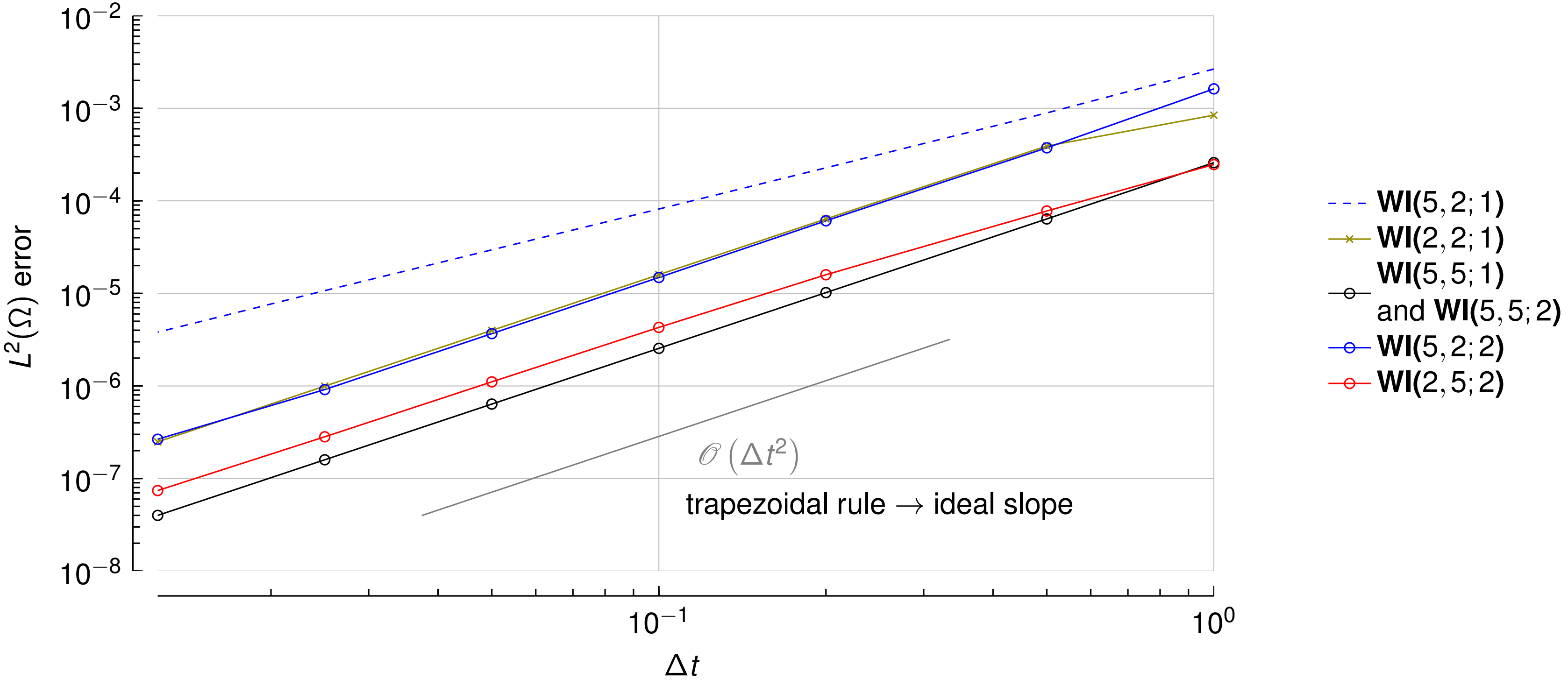
Partitioned Heat Equation



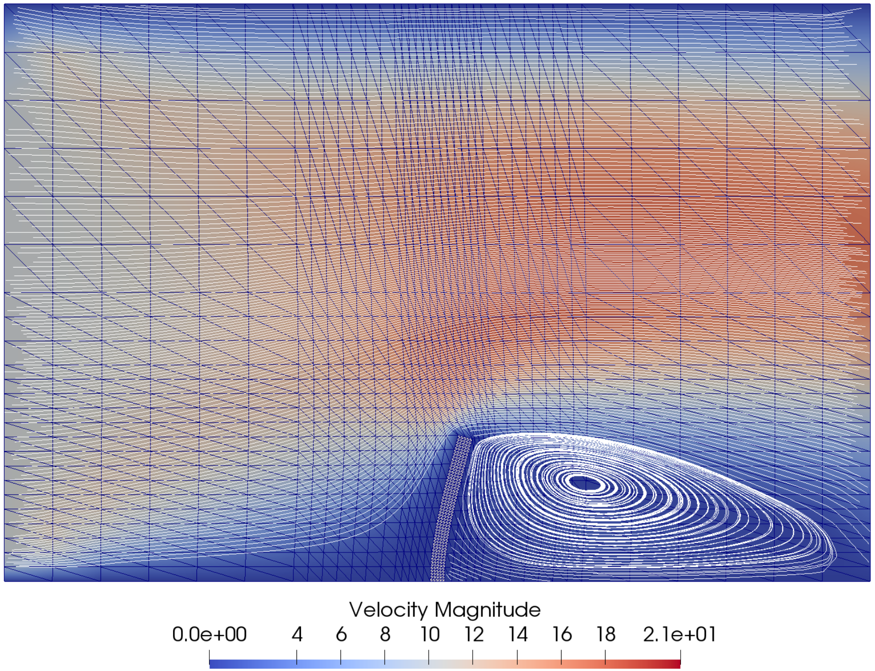
Partitioned Heat Equation



Partitioned Heat Equation

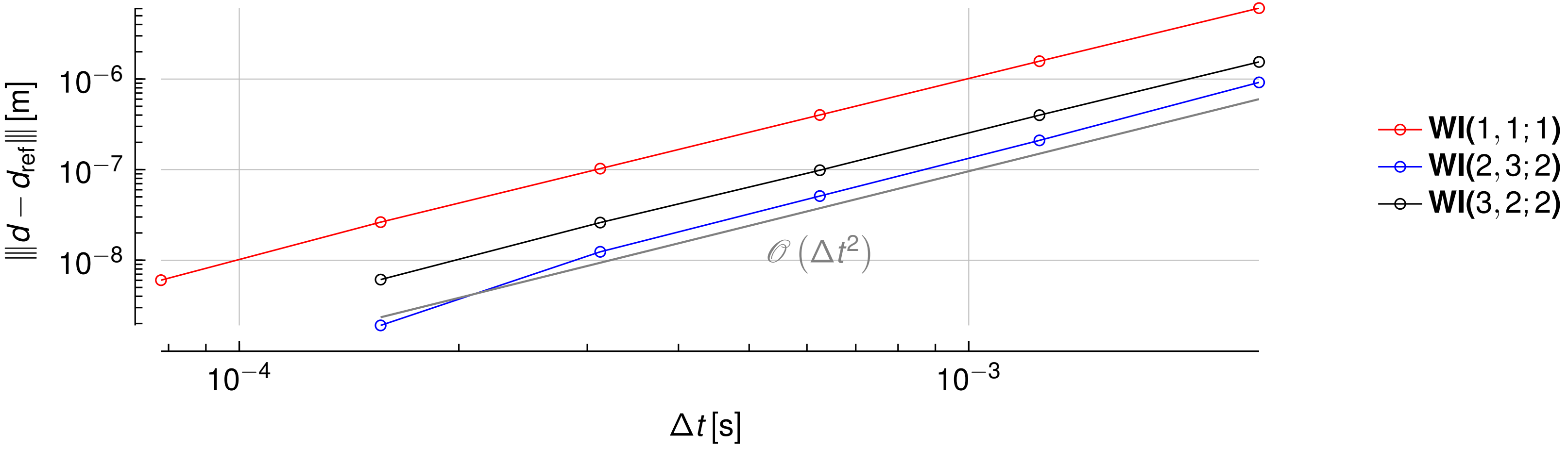


FSI Flap



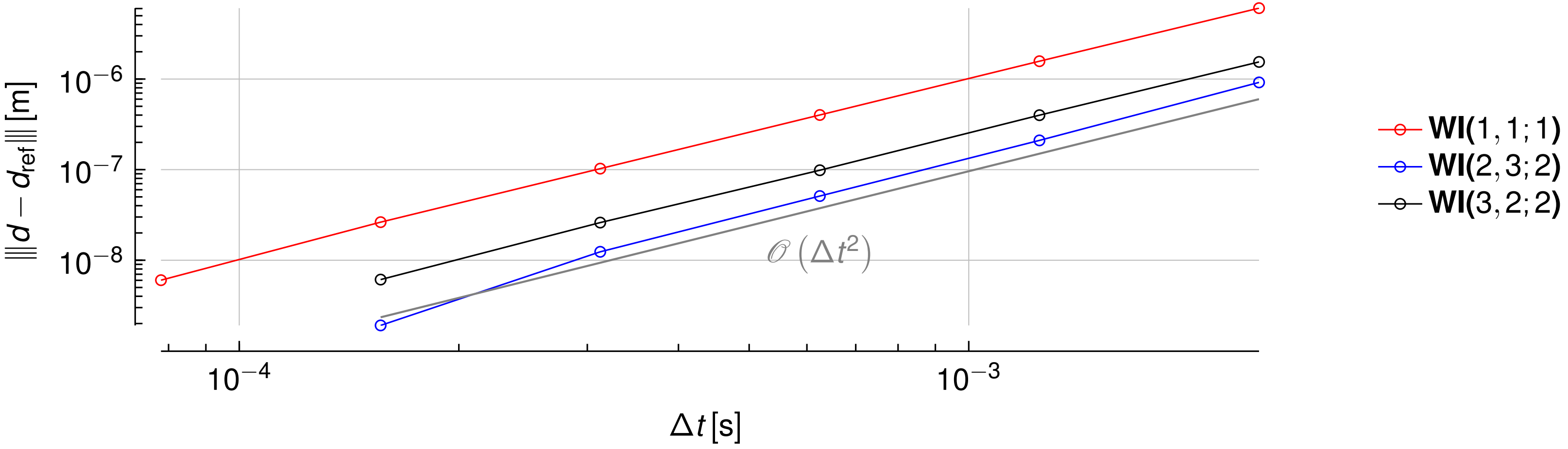
FSI Flap

Convergence study for trapezoidal rule (Fluid) and Newmark β method (Solid)



FSI Flap

Convergence study for trapezoidal rule (Fluid) and Newmark β method (Solid)



QN-Iterations

Δt_l[s]	$0.0025 \cdot 2^{0}$	$0.0025 \cdot 2^{-1}$	$0.0025 \cdot 2^{-2}$	$0.0025 \cdot 2^{-3}$	$0.0025 \cdot 2^{-4}$
WI(1, 1; 1)	4.00	4.50	4.81	5.50	5.64
WI(2, 3; 2)	5.25	5.63	6.57	7.31	7.42
WI(3, 2; 2)	4.50	4.75	5.31	5.63	6.33

Conclusion and future work

Conclusion

- $\delta t \neq \Delta t$
- partitioned black-box solvers can efficiently use multirate + QN
- higher order can be reached
- functionality can be hidden inside preCICE

```
<participant name="D">
  <write-data name="q1"/>
  <write-data name="q2"/>
  <read-data  name="T1"/>
  <read-data  name="T2"/>
  <read-data  name="T3"/>
  <read-data  name="T4"/>
  <read-data  name="T5"/>
</participant>
```

Conclusion and future work

Conclusion

- $\delta t \neq \Delta t$
- partitioned black-box solvers can efficiently use multirate + QN
- higher order can be reached
- functionality can be hidden inside preCICE

```
<participant name="D">
  <write-data name="q"/>
  <read-data  name="T"/>
</participant>
```

Future work

- real preCICE implementation
- explicit coupling + extrapolation

Conclusion and future work

```xml
<participant name="D">
  <write-data name="q"/>
  <read-data  name="T"/>
</participant>
```

Conclusion

- $\delta t \neq \Delta t$
- partitioned black-box solvers can efficiently use multirate + QN
- higher order can be reached
- functionality can be hidden inside preCICE

Future work

- real preCICE implementation
- explicit coupling + extrapolation

Interested in details?

Rüth, B., Uekermann, B., Mehl, M., Birken, P., Monge, A., & Bungartz, H. J. (2020). Quasi-Newton Waveform Iteration for Partitioned Fluid-Structure Interaction. arXiv preprint arXiv:2001.02654.