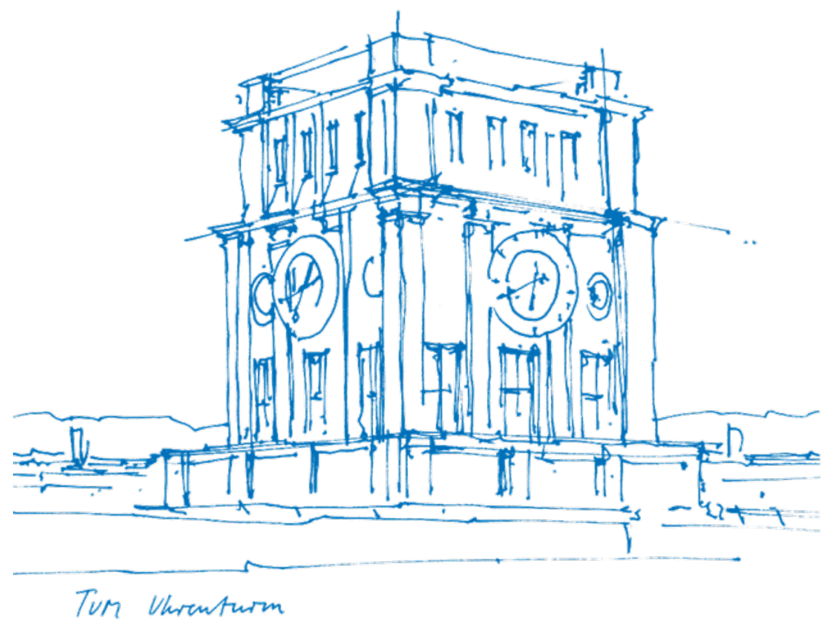
Solving the Partitioned Heat Equation Using FEniCS and preCICE

Benjamin Rüth[1], Peter Meisrimel[2], Philipp Birken[2], Benjamin Uekermann[1]

[1]Technical University of Munich
Department of Informatics
Chair of Scientific Computing

[2]Lund University
Mathematics (Faculty of Sciences)
Numerical Analysis



Siegen, Germany
November 29, 2018

Agenda

Partitioned Approach

Heat Equation with FEniCS

Coupling with preCICE

Results

Agenda

Partitioned Approach

Heat Equation with FEniCS

Coupling with preCICE

Results

A few Disclaimers:

This talk is **not**

- a talk about FEM
- a talk about coupling algorithms
- a talk with proper mathematical notation

Agenda

Partitioned Approach

Heat Equation with FEniCS

Coupling with preCICE

Results

A few Disclaimers:

This talk is **not**

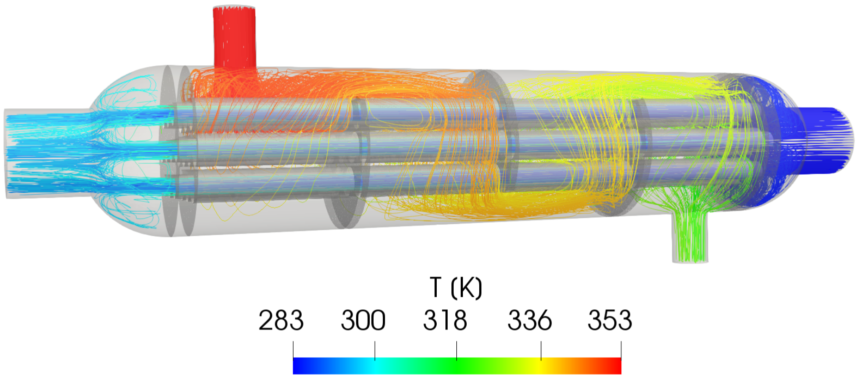
- a talk about FEM
- a talk about coupling algorithms
- a talk with proper mathematical notation

I will talk about

- software
- the partitioned approach
- where you can find my code
- how you can use my code

Partitioned Approach

Coupled problems

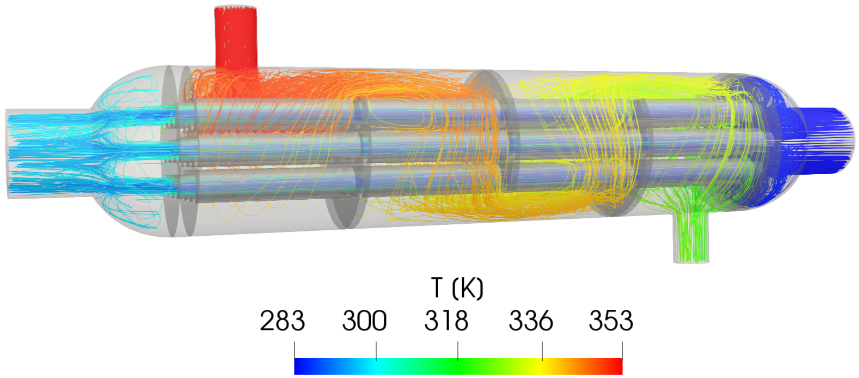


shell and tube heat exchanger using OpenFOAM and CalculiX[1]

[1] Figure from *Rusch, A., Uekermann, B. Comparing OpenFOAM's Intrinsic Conjugate Heat Transfer Solver with preCICE-Coupled Simulations. Technical Report, 2018.*

Partitioned Approach

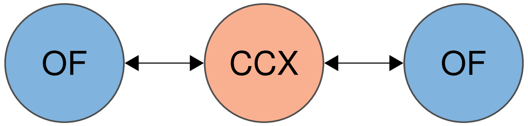
Coupled problems



shell and tube heat exchanger using OpenFOAM and CalculiX[1]

Basic idea:

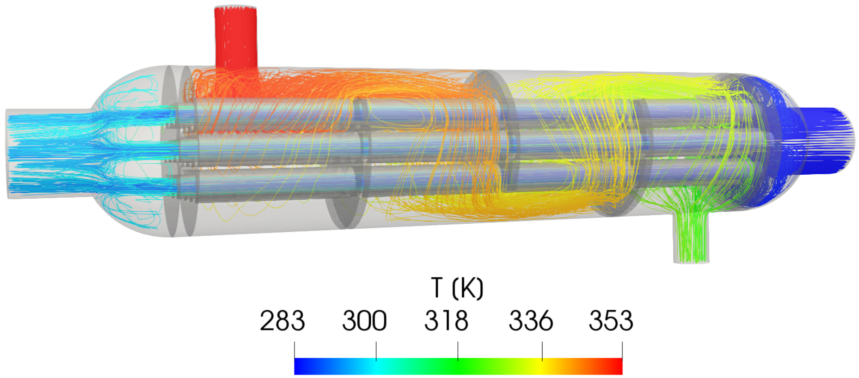
- reuse existing solvers
- combine single-physics to solve multi-physics
- only exchange "black-box" information



[1] Figure from *Rusch, A., Uekermann, B. Comparing OpenFOAM's Intrinsic Conjugate Heat Transfer Solver with preCICE-Coupled Simulations. Technical Report, 2018.*

Partitioned Approach

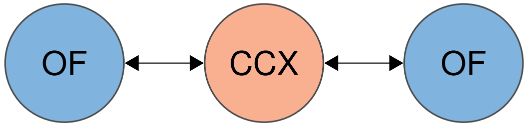
Coupled problems



shell and tube heat exchanger using OpenFOAM and CalculiX[1]

Basic idea:

- reuse existing solvers
- combine single-physics to solve multi-physics
- only exchange "black-box" information



What we do today:

- couple with preCICE library
- use FEniCS as a solver for toy problem

[1] Figure from *Rusch, A., Uekermann, B. Comparing OpenFOAM's Intrinsic Conjugate Heat Transfer Solver with preCICE-Coupled Simulations. Technical Report, 2018.*

Partitioned Approach

preCICE[1]

LUND UNIVERSITY

Features

- communication
- coupling schemes
- mapping
- time interpolation
- **official adapters** for OpenFOAM, SU2,...

preCICE

github.com/precice

---

[1] *Bungartz, H.-J., et al. (2016). preCICE – A fully parallel library for multi-physics surface coupling.*
[2] *Uekermann, B., et al. (2017). Official preCICE Adapters for Standard Open-Source Solvers.*

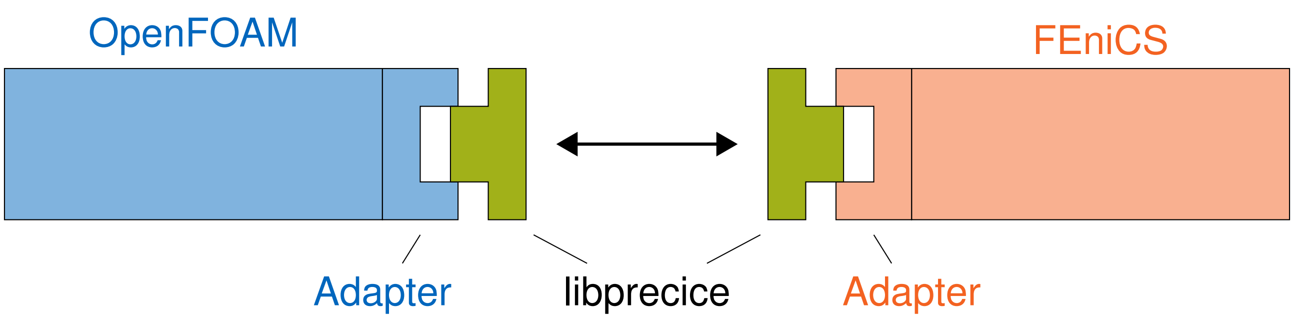
Partitioned Approach

preCICE[1]

Adapter[2]

- access preCICE API
- isolated layer between solver and preCICE
- support component exchangeability
- don't change existing (reliable, well-tested) code

github.com/precice



[1] *Bungartz, H.-J., et al. (2016). preCICE – A fully parallel library for multi-physics surface coupling.*
[2] *Uekermann, B., et al. (2017). Official preCICE Adapters for Standard Open-Source Solvers.*

Partitioned Approach

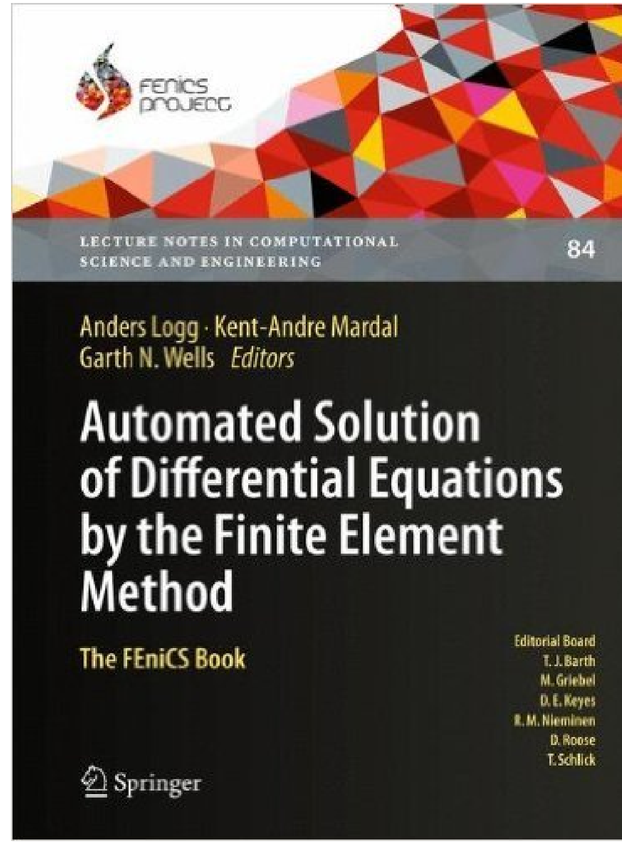
FEniCS[1]

Software

- open-source (LGPLv3)
- extensive documentation
- Python and C++ API
- can be used for HPC
- `www.fenicsproject.org`

Computing platform for solving PDEs

- Definition of weak forms
- Finite Element basis functions
- Meshing
- Solving
- ...



FEniCS book[2]

[1] *Alnaes, M. S., et al. (2015). The FEniCS Project Version 1.5.*

[2] *Logg, A., Mardal, K. A., & Wells, G. N. (2012). Automated solution of differential equations by the finite element method.*

Partitioned Approach

FEniCS[1]

Software

- open-source (LGPLv3)
- extensive documentation
- Python and C++ API
- can be used for HPC
- `www.fenicsproject.org`

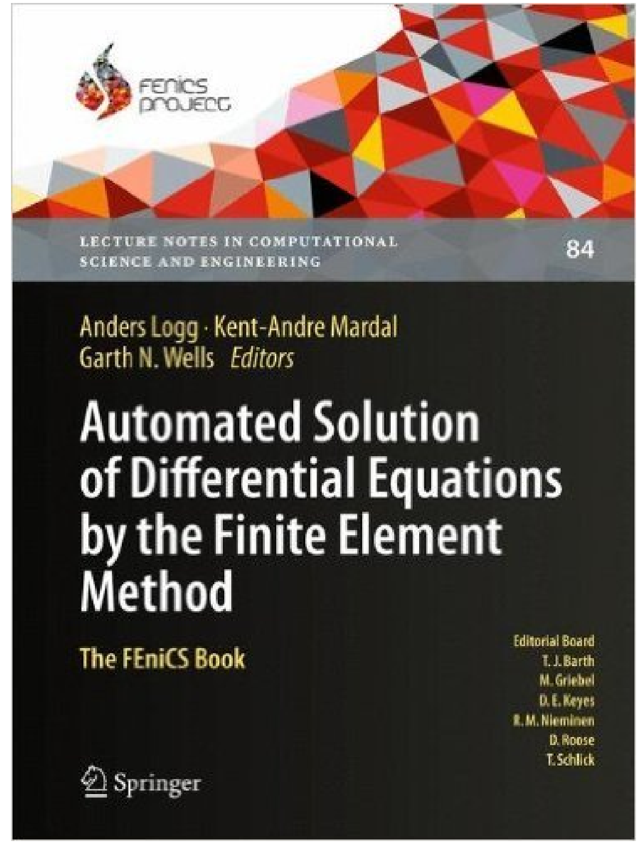
Computing platform for solving PDEs

- Definition of weak forms
- Finite Element basis functions
- Meshing
- Solving
- ...

→ You can do a lot of things with FEniCS!

My goal:

Develop an official preCICE adapter for FEniCS.



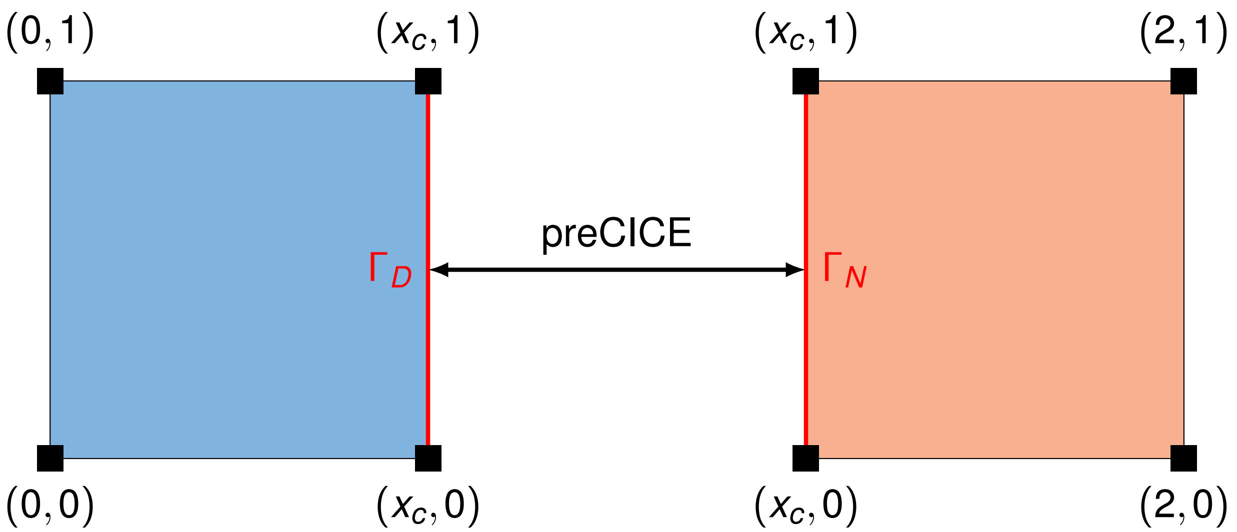
FEniCS book[2]

[1] *Alnaes, M. S., et al. (2015). The FEniCS Project Version 1.5.*

[2] *Logg, A., Mardal, K. A., & Wells, G. N. (2012). Automated solution of differential equations by the finite element method.*

Partitioned Approach

Toy problem: Partitioned Heat Equation



Partitioned heat equation / transmission problem already discussed in literature (e.g.[1] or [2]).

[1] Monge, A. (2018). Partitioned methods for time-dependent thermal fluid-structure interaction. Lund University.

[2] Toselli, A., & Widlund, O. (2005). Domain Decomposition Methods - Algorithms and Theory (1st ed.).

Heat Equation with FEniCS

Partitioned Heat Equation

get Ω_1: $u_D = u_1(x,y)$ set Ω_2: $u_2(x,y) = u_D$

Dirichlet BC

heat equation on Ω_1

$$\frac{\partial u_1}{\partial t} = \Delta u_1 + f$$

heat equation on Ω_2

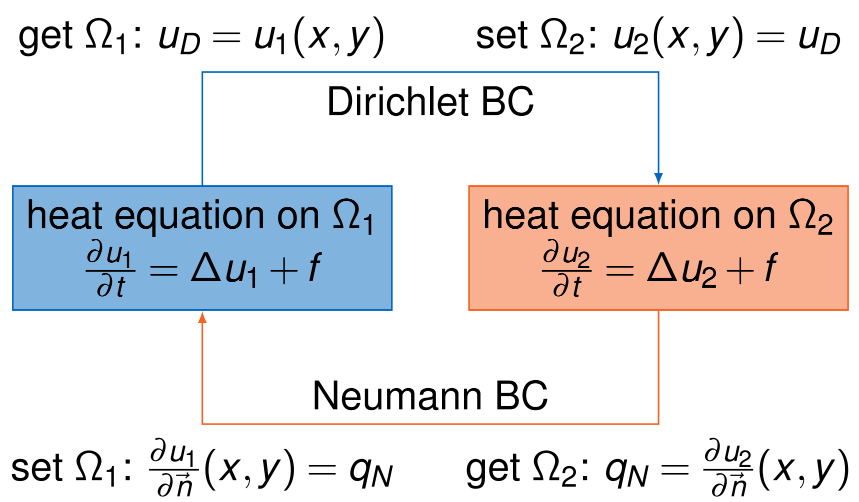
$$\frac{\partial u_2}{\partial t} = \Delta u_2 + f$$

Neumann BC

set Ω_1: $\frac{\partial u_1}{\partial \vec{n}}(x,y) = q_N$ get Ω_2: $q_N = \frac{\partial u_2}{\partial \vec{n}}(x,y)$

Heat Equation with FEniCS

Partitioned Heat Equation



FEniCS Ingredients

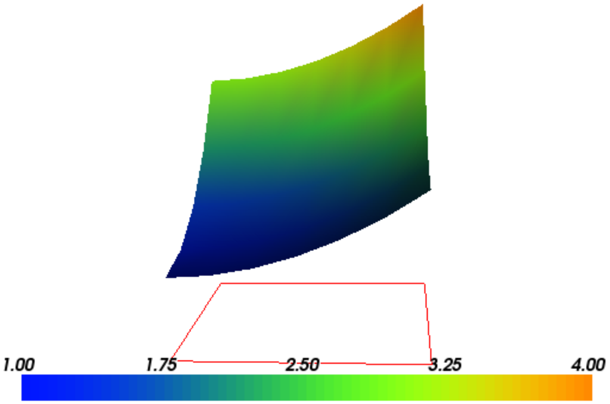
1. Solve Dirichlet Problem $\mathscr{D}(u_D)$
2. Compute heat flux $\mathscr{D}(u_D) = q_N$
3. Solve Neumann Problem $\mathscr{N}(q_N) = u_D$

Heat Equation with FEniCS

1. Solve Dirichlet Problem $\mathscr{D}(u_D)$

Heat Equation

$$\frac{\partial u}{\partial t} = \Delta u + f \text{ in } \Omega$$
$$u = u_0(t) \text{ on } \partial\Omega$$



Solution of Poisson equation. Figure from [1].

[1] Logg, A., Mardal, K. A., & Wells, G. N. (2012). Automated solution of differential equations by the finite element method.

Heat Equation with FEniCS

1. Solve Dirichlet Problem $\mathscr{D}(u_D)$

Discretization

- **implicit Euler:**

$$\frac{u^k - u^{k-1}}{dt} = \Delta u^k + f^k$$

- **trial space:**

$$u \in V_h \subset V = \left\{ v \in H^1(\Omega) : v = u_0 \text{ on } \partial\Omega \right\}$$

- **test space:**

$$v \in \hat{V}_h \subset V = \left\{ v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega \right\}$$

- **weak form:**

$$\int_\Omega (u^k v + \mathrm{d}t \nabla u^k \cdot \nabla v) \mathrm{d}x = \int_\Omega \left(u^{k-1} + \mathrm{d}t\, f^k\right) v \mathrm{d}x$$

[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.).*

Heat Equation with FEniCS

1. Solve Dirichlet Problem $\mathscr{D}(u_D)$

Discretization

- **implicit Euler:**

$$\frac{u^k - u^{k-1}}{dt} = \Delta u^k + f^k$$

- **trial space:**

$$u \in V_h \subset V = \left\{ v \in H^1(\Omega) : v = u_0 \text{ on } \partial\Omega \right\}$$

- **test space:**

$$v \in \hat{V}_h \subset V = \left\{ v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega \right\}$$

- **weak form:**

$$\int_\Omega (u^k v + \mathrm{d}t \nabla u^k \cdot \nabla v) \mathrm{d}x = \int_\Omega \left(u^{k-1} + \mathrm{d}t\, f^k \right) v \mathrm{d}x$$

Analytical Solution

If right-hand-side $f = \beta - 2 - 2\alpha$ we get $u = 1 + x^2 + \alpha y^2 + \beta t$.

[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.).*

Heat Equation with FEniCS

1. Solve Dirichlet Problem $\mathscr{D}(u_D)$

Discretization

- **implicit Euler:**

$$\frac{u^k - u^{k-1}}{dt} = \Delta u^k + f^k$$

- **trial space:**

$$u \in V_h \subset V = \left\{ v \in H^1(\Omega) : v = u_0 \text{ on } \partial\Omega \right\}$$

- **test space:**

$$v \in \hat{V}_h \subset V = \left\{ v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega \right\}$$

- **weak form:**

$$\int_\Omega (u^k v + \mathrm{d}t \nabla u^k \cdot \nabla v) \mathrm{d}x = \int_\Omega \left(u^{k-1} + \mathrm{d}t\, f^k \right) v \mathrm{d}x$$

Analytical Solution

If right-hand-side $f = \beta - 2 - 2\alpha$ we get $u = 1 + x^2 + \alpha y^2 + \beta t$.

weak form in FEniCS

```
F = u*v*dx + dt*dot(grad(u),grad(v))*dx - (u_n+dt*f)*v*dx
```

Remark: Tutorial from the FEniCS tutorial book[1]

[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.).*

Heat Equation with FEniCS

2. Compute heat flux $\mathscr{D}(u_D) = q_N$

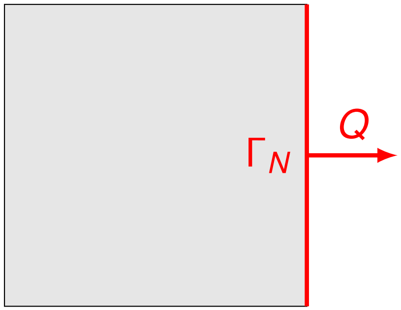
Overall Heat Flux

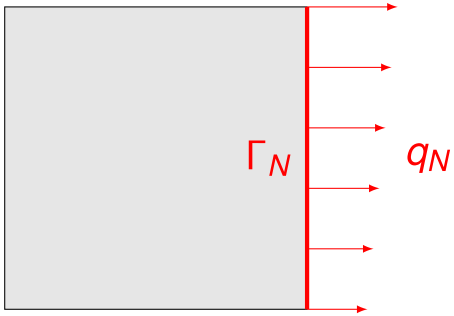
$$Q = -K \int_{\Gamma_N} \frac{\partial u}{\partial \vec{n}} \mathrm{d}s \quad (K: \text{Thermal Conductivity})$$

Elementwise Heat Flux[1]

$$\mu_i^k = \int_{\Gamma_N} \frac{\partial u^k}{\partial \vec{n}} v_i \mathrm{d}s = \int_{\Omega} u^k v_i - u^{k-1} v_i + \mathrm{d}t \nabla u^k \cdot \nabla v_i - \mathrm{d}t f^k v_i \mathrm{d}x$$

$$q_N = -K \sum_i v_i \mu_i^k$$





[1] Toselli, A., & Widlund, O. (2005). Domain Decomposition Methods - Algorithms and Theory (1st ed.). p.3 f.

Heat Equation with FEniCS

3. Solve Neumann Problem $\mathscr{N}(q_N) = u_D$

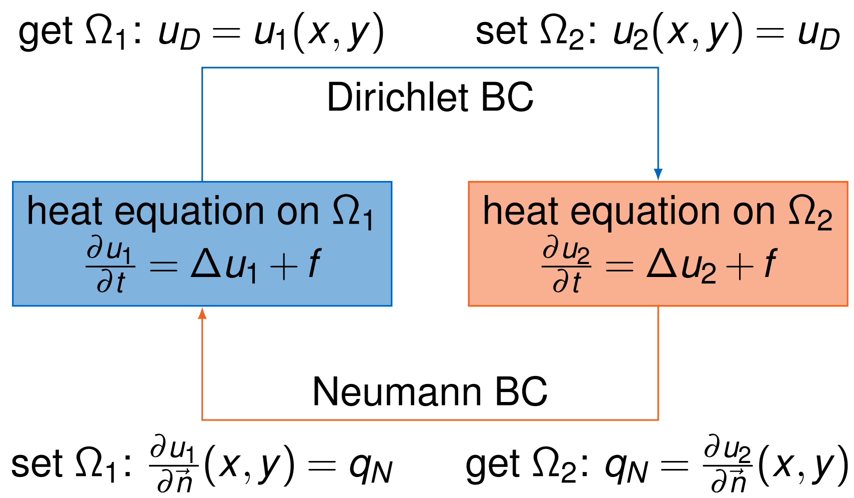
Neumann BC: Modified weak form

$$\int_\Omega (u^k v + \mathrm{d}t \nabla u^k \cdot \nabla v)\mathrm{d}x = \int_\Omega \left(u^{k-1} + \mathrm{d}t\, f^k\right) v\mathrm{d}x + \int_{\Gamma_N} q_N v\,\mathrm{d}s$$

LUND UNIVERSITY

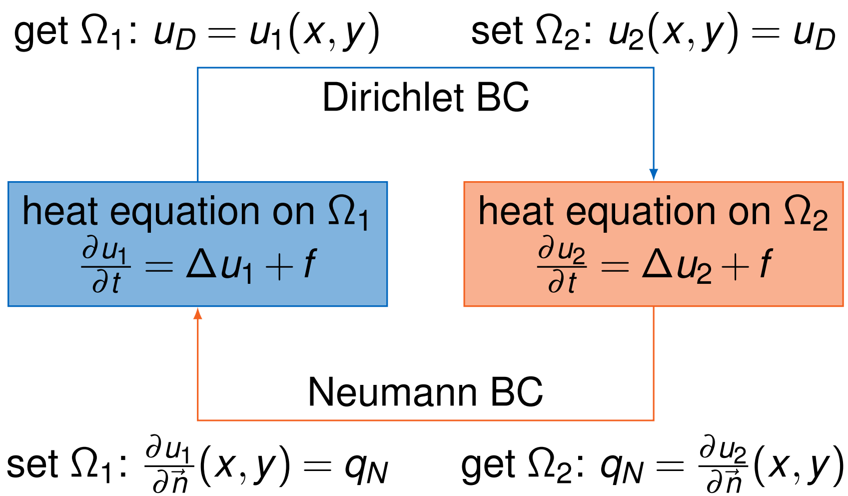
TUM

Partitioned Heat Equation



Coupling with preCICE

Partitioned Heat Equation



preCICE Ingredients

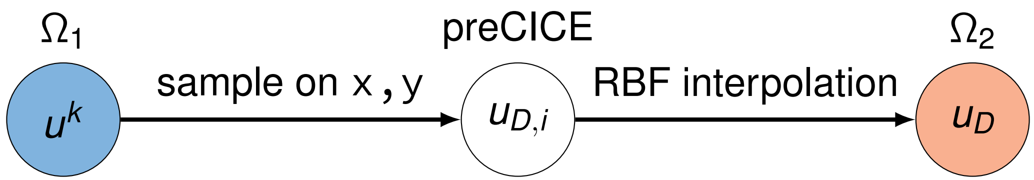
1. Read coupling data u_D, q_N to nodal data $u_{D,i}, q_{N,i}$
2. Apply coupling boundary conditions u_D, q_N
3. preCICE-FEniCS Adapter

Coupling with preCICE
1. Read Coupling Data

- Read Temperature u_D: `u_np1(x, y)`
- Read Flux q_N: `fluxes(x, y)`
- preCICE only accepts **nodal data** on the **coupling mesh**

Coupling with preCICE

2. Apply Coupling Boundary Conditions



- preCICE only returns **nodal data** on the **coupling mesh**
- use RBF interpolation to create a `CustomExpression(UserExpression)`
- Write Flux q_N as Neumann BC
- Write Temperature u_D as Dirichlet BC

Coupling with preCICE

3. preCICE-FEniCS Adapter

Example usage from the perspective of the Dirichlet solver $\mathscr{D}(u_D) = q_N$

```
from fenics import *
from fenicsadapter import Adapter
...
adapter = Adapter()
```

Coupling with preCICE

3. preCICE-FEniCS Adapter

Example usage from the perspective of the Dirichlet solver $\mathscr{D}(u_D) = q_N$

```
from fenics import *
from fenicsadapter import Adapter
...
adapter = Adapter()
adapter.configure("HeatDirichlet", "precice_config.xml",
                  "DirichletNodes", "Flux", "Temperature")
adapter.initialize(coupling_boundary, mesh, f_N_function, u_D_function)
```

Coupling with preCICE

3. preCICE-FEniCS Adapter

Example usage from the perspective of the Dirichlet solver $\mathscr{D}(u_D) = q_N$

```
from fenics import *
from fenicsadapter import Adapter
...
adapter = Adapter()
adapter.configure("HeatDirichlet", "precice_config.xml",
                  "DirichletNodes", "Flux", "Temperature")
adapter.initialize(coupling_boundary, mesh, f_N_function, u_D_function)
...
bcs = [DirichletBC(V, u_D, remaining_boundary)]
bcs.append(adapter.create_coupling_dirichlet_boundary_condition(V))
```

LUND UNIVERSITY

Example usage from the perspective of the Dirichlet solver $\mathscr{D}(u_D) = q_N$

```
from fenics import *
from fenicsadapter import Adapter
...
adapter = Adapter()
adapter.configure("HeatDirichlet", "precice_config.xml",
                  "DirichletNodes", "Flux", "Temperature")
adapter.initialize(coupling_boundary, mesh, f_N_function, u_D_function)
...
bcs = [DirichletBC(V, u_D, remaining_boundary)]
bcs.append(adapter.create_coupling_dirichlet_boundary_condition(V))
...
while adapter.is_coupling_ongoing():
    solve(a == L, u_np1, bcs)
    fluxes = fluxes_from_temperature(F, V)
```

Coupling with preCICE

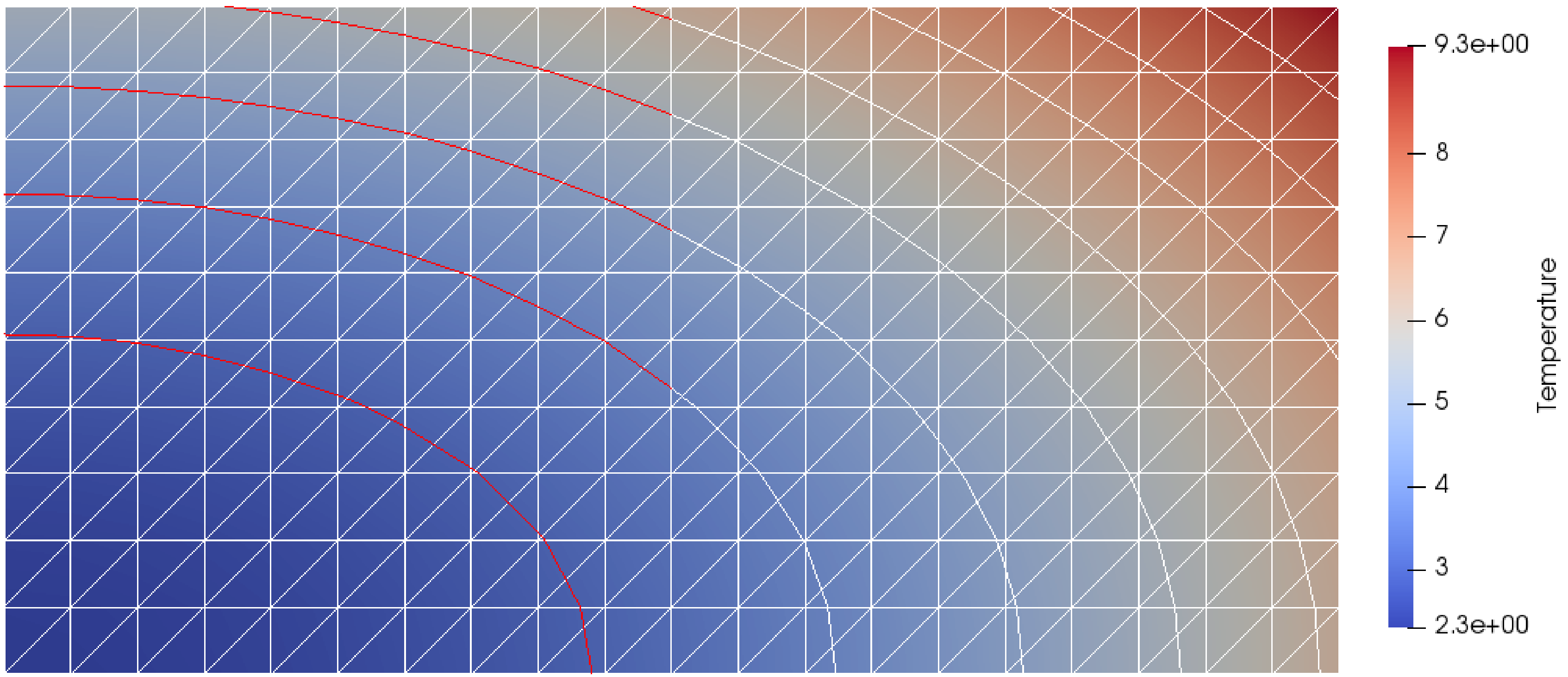
3. preCICE-FEniCS Adapter

Example usage from the perspective of the Dirichlet solver $\mathscr{D}(u_D) = q_N$

```
from fenics import *
from fenicsadapter import Adapter
...
adapter = Adapter()
adapter.configure("HeatDirichlet", "precice_config.xml",
                  "DirichletNodes", "Flux", "Temperature")
adapter.initialize(coupling_boundary, mesh, f_N_function, u_D_function)
...
bcs = [DirichletBC(V, u_D, remaining_boundary)]
bcs.append(adapter.create_coupling_dirichlet_boundary_condition(V))
...
while adapter.is_coupling_ongoing():
    solve(a == L, u_np1, bcs)
    fluxes = fluxes_from_temperature(F, V)
    is_converged = adapter.advance(fluxes, dt)
    if is_converged:
        ...
```

Results

Matching meshes

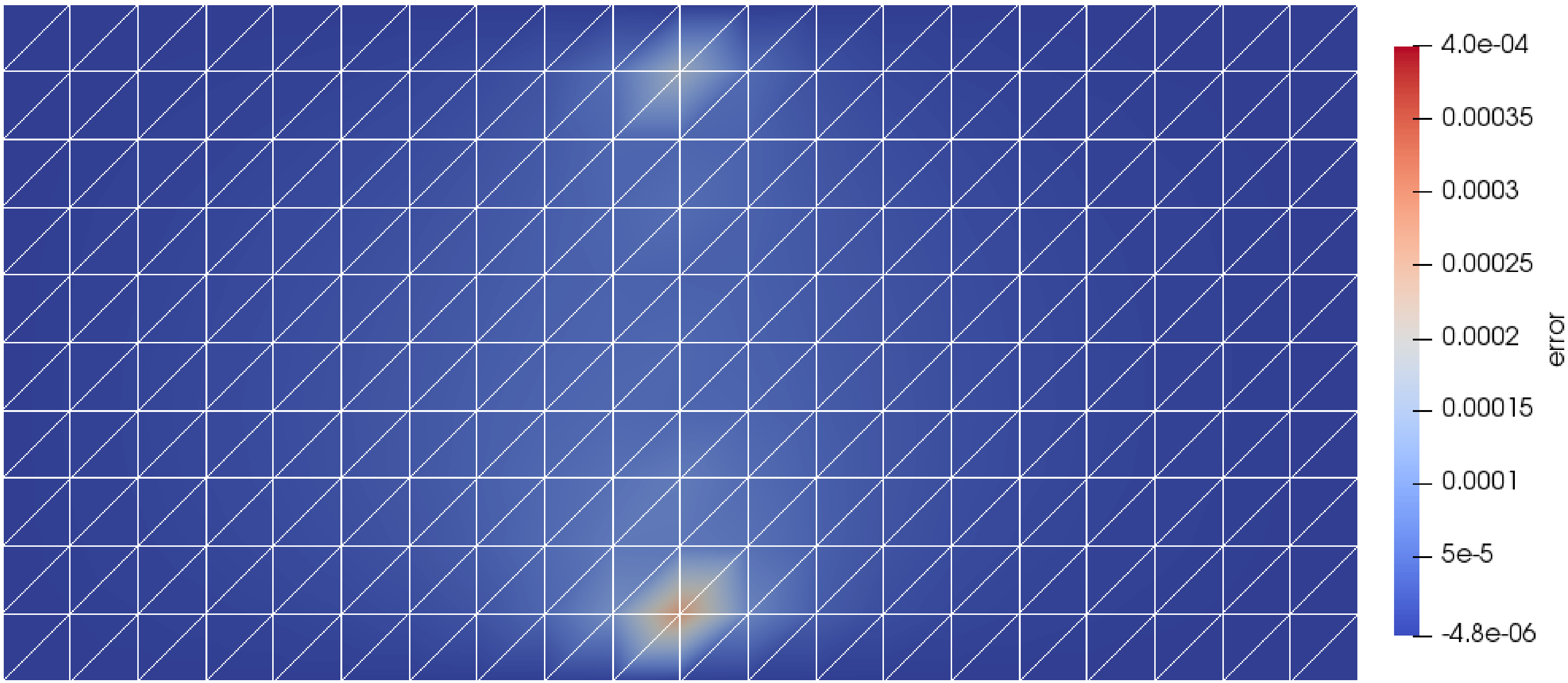


Comments

- simple heat equation from above
- "eyeball norm:" agreement with monolithic and analytical solution $u = 1 + x^2 + \alpha y^2 + \beta t$

Results

Matching meshes

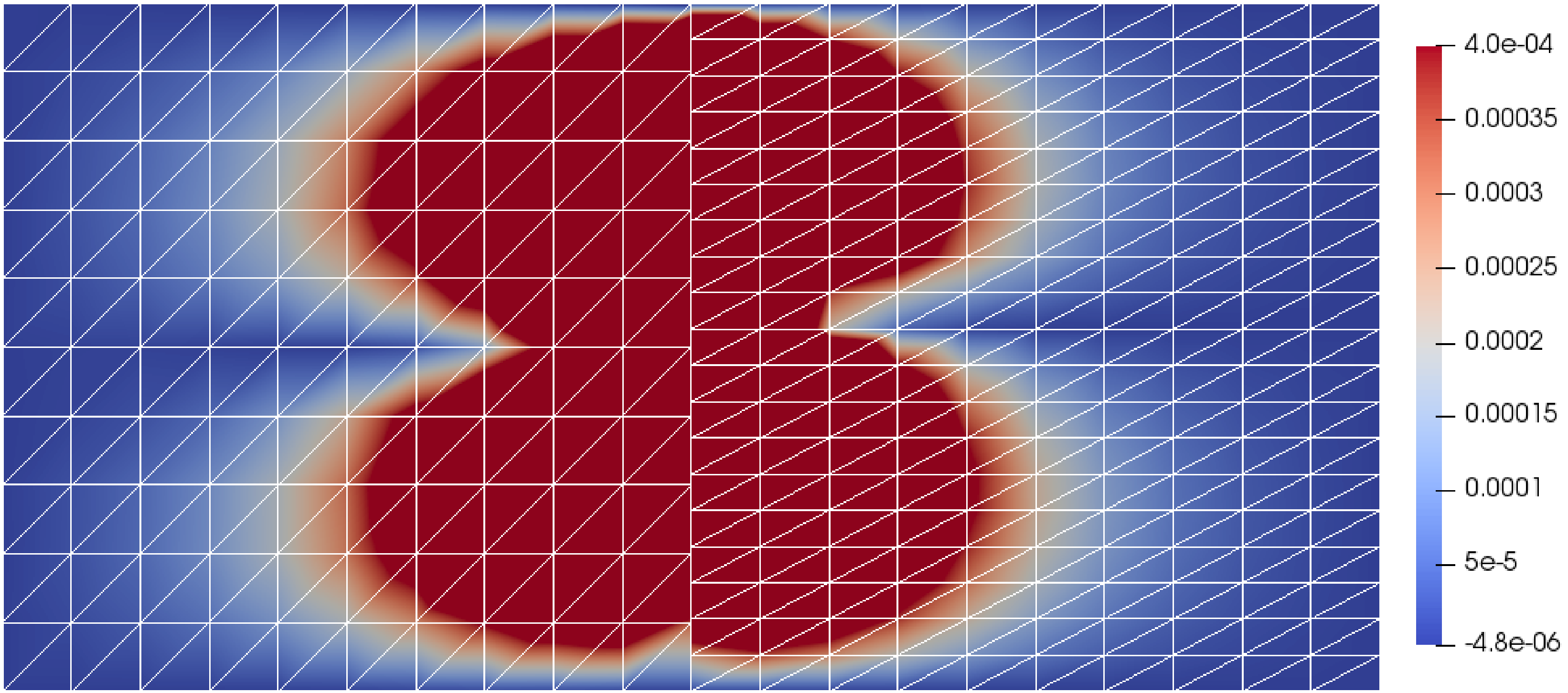


Comments

- simple heat equation from above
- "eyeball norm:" agreement with monolithic and analytical solution $u = 1 + x^2 + \alpha y^2 + \beta t$
- L^2-error on domain $< 10^{-4}$

Results

Non-matching meshes



Comments

- simple heat equation from above
- finer mesh on right domain, but larger error
- possible explanation: first order mapping destroys second order accuracy of space discretization

Summary & Outlook

Partitioned heat equation

- FEniCS is used for solving the Dirichlet and Neumann problem.
- preCICE couples two FEniCS instances to solve the coupled problem.

[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.). Sec. 3.1*

Summary & Outlook

Partitioned heat equation

- FEniCS is used for solving the Dirichlet and Neumann problem.
- preCICE couples two FEniCS instances to solve the coupled problem.

FEniCS adapter

- only minimal changes in the official FEniCS tutorial for the heat equation[1].
- FEniCS adapter for heat transport
- `github.com/precice/fenics-adapter`
- `github.com/precice/tutorials`

[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.). Sec. 3.1*

Summary & Outlook

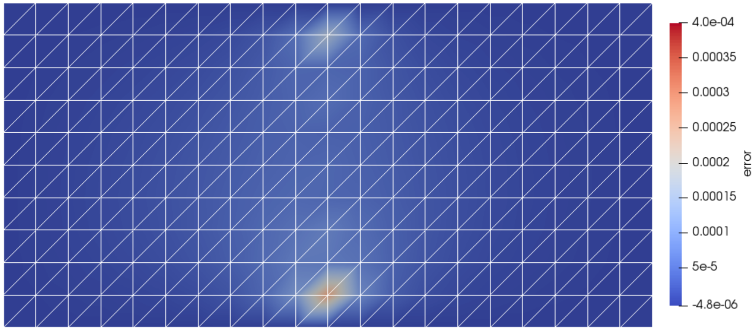
Partitioned heat equation

- FEniCS is used for solving the Dirichlet and Neumann problem.
- preCICE couples two FEniCS instances to solve the coupled problem.

FEniCS adapter

- only minimal changes in the official FEniCS tutorial for the heat equation[1].
- FEniCS adapter for heat transport
- `github.com/precice/fenics-adapter`
- `github.com/precice/tutorials`

Can we live with the error close to the boundary?

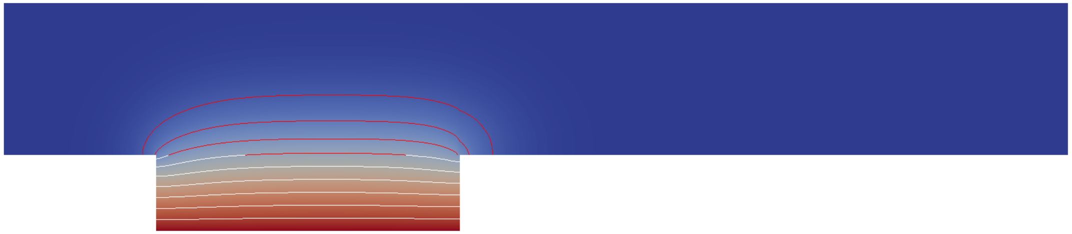


[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.). Sec. 3.1*

Summary & Outlook

Outlook: FEniCS + X

- first experiments with FEniCS + OpenFOAM
- more FEniCS tutorials
- FEniCS based solvers as CBC.Block, CBC.RANS and CBC.Solve[1]



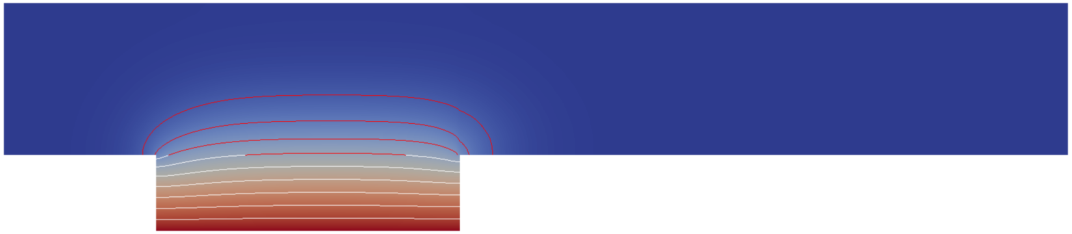
Flow over heated plate. FEniCS used for solving the heat equation inside the hot plate at the bottom and OpenFOAM used for simulation of the channel flow.

[1] *Logg, A., Mardal, K. A., & Wells, G. N. (2012). Automated solution of differential equations by the finite element method. Lecture Notes in Computational Science and Engineering.*

Summary & Outlook

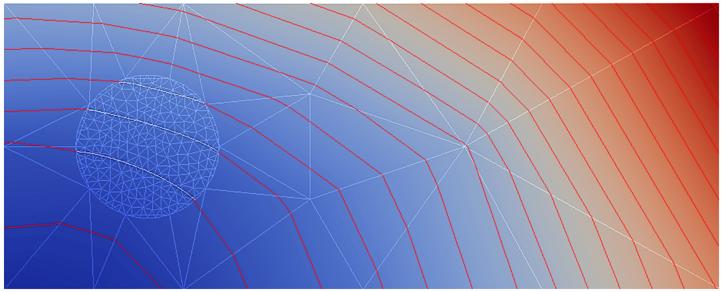
Outlook: FEniCS + X

- first experiments with FEniCS + OpenFOAM
- more FEniCS tutorials
- FEniCS based solvers as CBC.Block, CBC.RANS and CBC.Solve[1]



Flow over heated plate. FEniCS used for solving the heat equation inside the hot plate at the bottom and OpenFOAM used for simulation of the channel flow.

Non-matching Meshes



[1] *Logg, A., Mardal, K. A., & Wells, G. N. (2012). Automated solution of differential equations by the finite element method. Lecture Notes in Computational Science and Engineering.*

Thank You!

Website: `precice.org`
Source/Wiki: `github.com/precice`
Mailing list: `precice.org/resources`
My e-mail: `rueth@in.tum.de`

Homework:

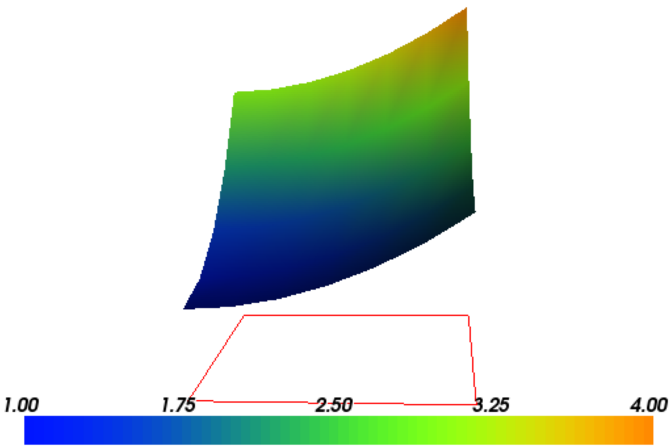
- Follow a tutorial
- Join our mailing list
- Star on GitHub
- Send us feedback
- Ask me for stickers

Dirichlet Problem with FEniCS

Heat Equation

$$\frac{\partial u}{\partial t} = \Delta u + f \text{ in } \Omega$$
$$u = u_0(t) \text{ on } \partial\Omega$$

Analytical Solution, if $f = \beta - 2 - 2\alpha$ we get $u = 1 + x^2 + \alpha y^2 + \beta t$.



Solution of Poisson equation. Figure from [1].

Discretization

- **implicit Euler:**

$$\frac{u^k - u^{k-1}}{dt} = \Delta u^k + f^k$$

- **trial space:**

$$u \in V_h \subset V = \left\{ v \in H^1(\Omega) : v = u_0 \text{ on } \partial\Omega \right\}$$

- **test space:**

$$v \in \hat{V}_h \subset V = \left\{ v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega \right\}$$

- **weak form:**

$$\int_\Omega (u^k v + \mathrm{d}t \nabla u^k \cdot \nabla v) \mathrm{d}x = \int_\Omega \left(u^{k-1} + \mathrm{d}t\, f^k \right) v \mathrm{d}x$$

Remark: Tutorial from the FEniCS book[1]

[1] *Langtangen, H. P., & Logg, A. (2016). Solving PDEs in Python - The FEniCS Tutorial I (1st ed.).*

Dirichlet Problem with FEniCS

LUND UNIVERSITY

Geometry: $\Omega, \partial\Omega, \Gamma_D, \Gamma_N$

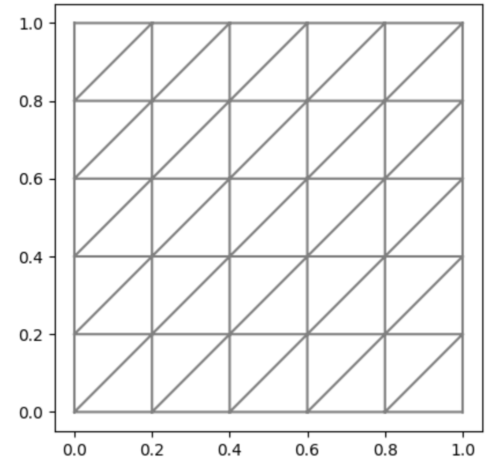
```
class RightBoundary(SubDomain):
    def inside(self, x, on_boundary):
        tol = 1E-14
        if on_boundary
           and near(x[0], x_r, tol):
            return True
        else:
            return False

class Boundary(SubDomain):
    def inside(self, x, on_boundary):
        if on_boundary:
            return True
        else:
            return False

p0 = Point(0, 0)
p1 = Point(1, 1)
```

Mesh: Ω_h

```
nx = 5
ny = 5
mesh = RectangleMesh(p0, p1,
                     nx, ny)
```



Mesh created with FEniCS

Dirichlet Problem with FEniCS

LUND UNIVERSITY

Function Space: $V_h \subset V = \left\{ v \in H^1(\Omega) \right\}$

```
V = FunctionSpace(mesh, 'P', 1)
```

Expressions: $u = 1 + x^2 + \alpha y^2 + \beta t$ and $f = \beta - 2 - 2\alpha$

```
u_D = Expression('1 + x[0]*x[0] + alpha*x[1]*x[1] + beta*t', ..., t=0)
f = Constant(beta - 2 - 2 * alpha)
```

Boundary Conditions: $u \in V_h \subset V = \left\{ v \in H^1(\Omega) : v = u_D \text{ on } \partial\Omega \right\}$ and $v \in \hat{V}_h \subset V = \left\{ v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega \right\}$

```
bc = DirichletBC(V, u_D, Boundary)
u = TrialFunction(V)
v = TestFunction(V)
```

Initial Condition: $u^0 = u(t = 0)$

```
u_n = interpolate(u_D, V)
```

Dirichlet Problem with FEniCS

Variational Problem: $\int_\Omega (u^k v + dt \nabla u^k \cdot \nabla v) dx = \int_\Omega \left(u^{k-1} + dt f^k\right) v dx$

```
F = u * v * dx + dt * dot(grad(u), grad(v)) * dx - (u_n + dt * f) * v *
    dx
a, L = lhs(F), rhs(F)
```

Time-stepping and simulation loop: $\frac{u^k - u^{k-1}}{dt} = \Delta u^k + f^k$

```
u_np1 = Function(V)
t = 0
T = 1
dt = .1
u_D.t = t + dt

while t < T:
    solve(a == L, u_np1, bc)
    t += dt
    u_D.t = t + dt
    u_n.assign(u_np1)
```