Theory and Practice of Finite Element Methods

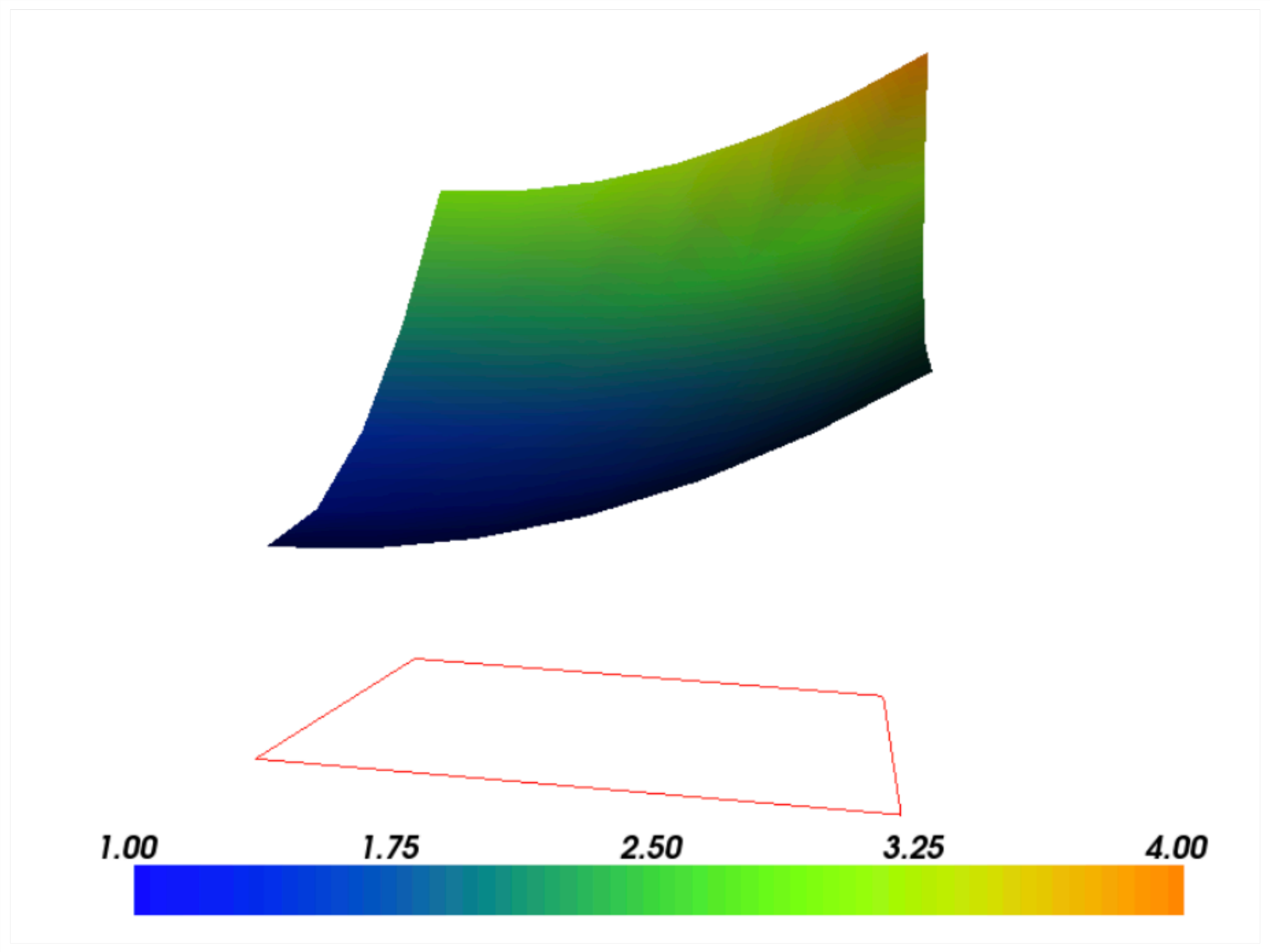
Luca Heltai <luca.heltai@sissa.it>

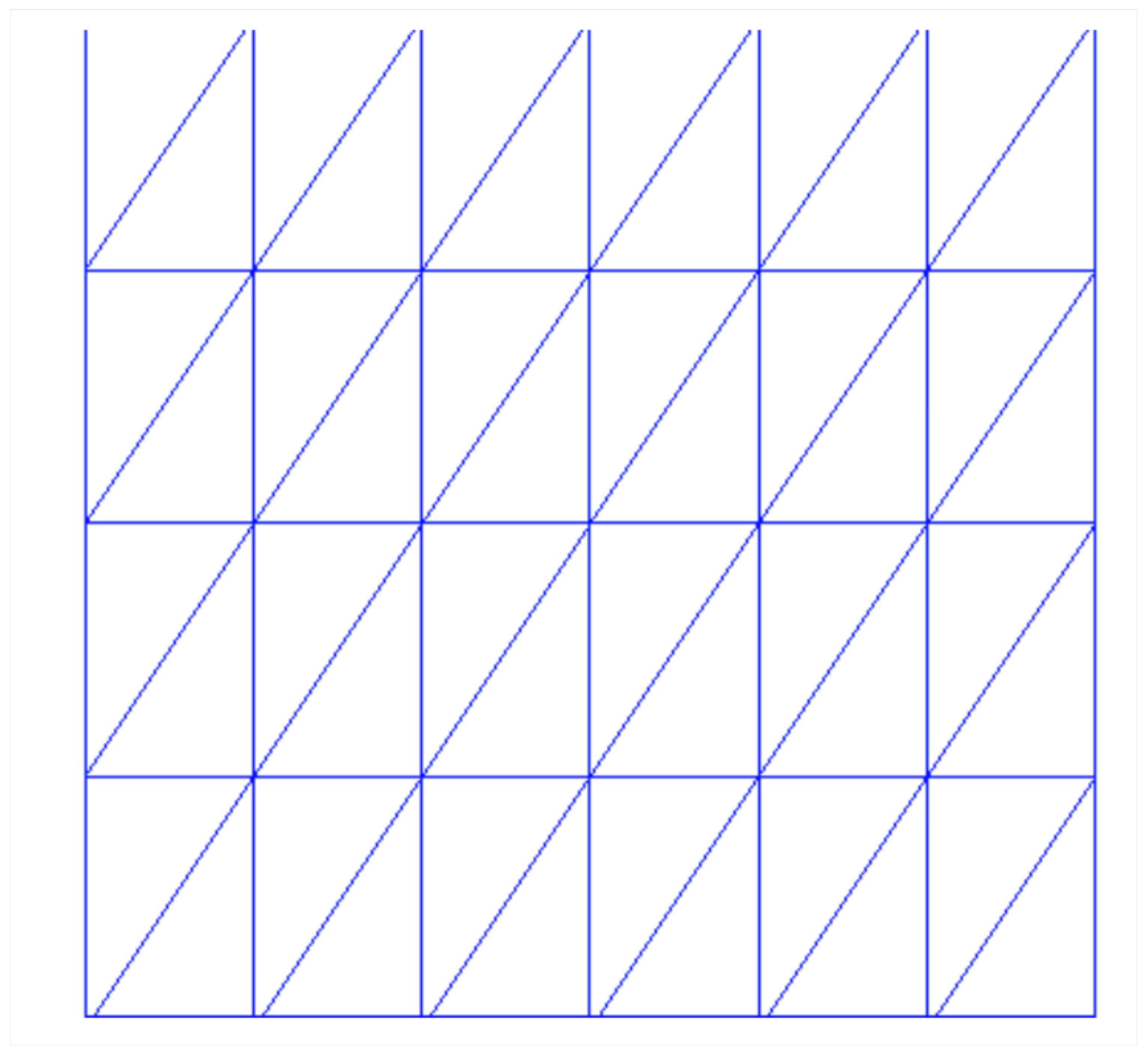
International School for Advanced Studies (www.sissa.it)
Mathematical Analysis, Modeling, and Applications (math.sissa.it)
Master in High Performance Computing (www.mhpc.it)
SISSA mathLab (mathlab.sissa.it)

Course objectives

```python
from fenics import *
import matplotlib.pyplot as plt

# Create mesh and define function space
mesh = UnitSquareMesh(8, 8)
V = FunctionSpace(mesh, 'P', 1)

# Define boundary condition
u_D = Expression('1 + x[0]*x[0] + 2*x[1]*x[1]', degree=2)

def boundary(x, on_boundary):
    return on_boundary

bc = DirichletBC(V, u_D, boundary)

# Define variational problem
u = TrialFunction(V)
v = TestFunction(V)
f = Constant(-6.0)
a = dot(grad(u), grad(v))*dx
L = f*v*dx

# Compute solution
u = Function(V)
solve(a == L, u, bc)

# Plot solution and mesh
plot(u)
plot(mesh)
```

Go from this:





Course objectives

To this:

Movie credit: Michał Wichrowski
Institute of Fundamental Technological Research

Two parallel parts

- Theory of Finite Elements (Tuesdays)
  - Galerkin methods
  - Basic theory: Lax Milgram, Cea, Bramble Hilbert, Nitsche’s trick, etc.
  - Petrov Galerkin methods
  - Mixed and Hybrid Methods
  - A posteriori error analysis

- Practice of Finite Elements (Thursdays)
  - Serial scalar poisson solver, in various flavours
  - Convergence tests
  - Vector and Mixed problems
  - Local adaptivity
  - Parallelisation techniques in FEM
  - If time permits: more advanced topics

Tools, Techniques, Best Practices

- What you will learn:
  - Advanced Finite Element theory
  - How to use a modern C++ IDE, to build and debug your codes
  - How to use a large FEM library to solve complex PDE problems
  - How to properly document your code using Doxygen
  - How to use a proper Git workflow to develop your applications
  - How to leverage GitHub actions, google tests, and docker images to test and deploy your application
  - How hybrid parallelisation (threads + MPI + GPU) works in real life FEM applications

Outcome of the course

- You will produce your own FEM application based on deal.II which:
  - Solves a PDE of interest to you, on adaptively refined grids, in parallel
  - Uses modern version control tools (on GitHub)
  - Is tested automatically (through GitHub actions) every time you push a commit, or open a pull request
  - Is documented using Doxygen, and its web page is updated and deployed automatically every time you merge to master a new branch

Prerequisites

- Theory:
  - Some knowledge of Sobolev Spaces
    - Linear operators, Banach and Hilbert spaces, duality, etc.
  - One elementary course on Numerical Analysis
    - Quadrature, interpolation, Taylor expansions, etc.
- Practice (for the first few lectures):
  - a machine with Visual Studio Code installed (c++-11 is required)
  - Docker
  - A GitHub account

More Info

- Course pages:
  - Official course page (the most up to date information is found here):
    https://www.math.sissa.it/course/phd-course/theory-and-practice-finite-element-methods
  - Course classroom on GitHub (for assignments/exercises)
    https://classroom.github.com/classrooms/14195552-theory-and-practice-of-finite-element-methods
  - Course slides, notes, materials, and codes:
    https://github.com/luca-heltai/theory-and-practice-of-fem
  - Course recordings:
    https://bit.ly/3uKDQWa
  - Email:
    prof. Luca Heltai <luca.heltai@sissa.it>

sissa mathLab innovating with mathematics

The Abdus Salam International Centre for Theoretical Physics

SISSA

Master in High Performance Computing

Theory and Practice of Finite Element Methods

Introduction

Luca Heltai <luca.heltai@sissa.it>

International School for Advanced Studies (www.sissa.it)
Mathematical Analysis, Modeling, and Applications (math.sissa.it)
Master in High Performance Computing (www.mhpc.it)
SISSA mathLab (mathlab.sissa.it)