Introduction to RTG's Software Platform deal.II

Denis Davydov (denis.davydov@fau.de)
Jean-Paul Pelteret (jean-paul.pelteret@fau.de)

10 May 2019

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Course goals

- Learn the fundamentals of deal.II
  - Commonly used data structures, their interface
  - Structure of finite element problems
  - Good implementation practices
- Solution of a Laplace equation using deal.II
- Brief introduction to local mesh refinement and hanging nodes
- Parallelization from the user perspective of deal.II
  - Shared memory and multithreading
  - Distributed memory

Course schedule

Time	Duration	Content	Speaker
09:00	45 min	Introduction to deal.II and Getting started	DD, JP
10:00	45 min	Solving Poisson's equation	JP
11:00	45 min	Local error estimation and adaptive refinement	DD
		Lunch break	
13:00	45 min	Shared memory parallelization	JP
14:00	45 min	MPI parallelization	DD
15:00	45 min	Code example and Q&A	JP, DD

Resources

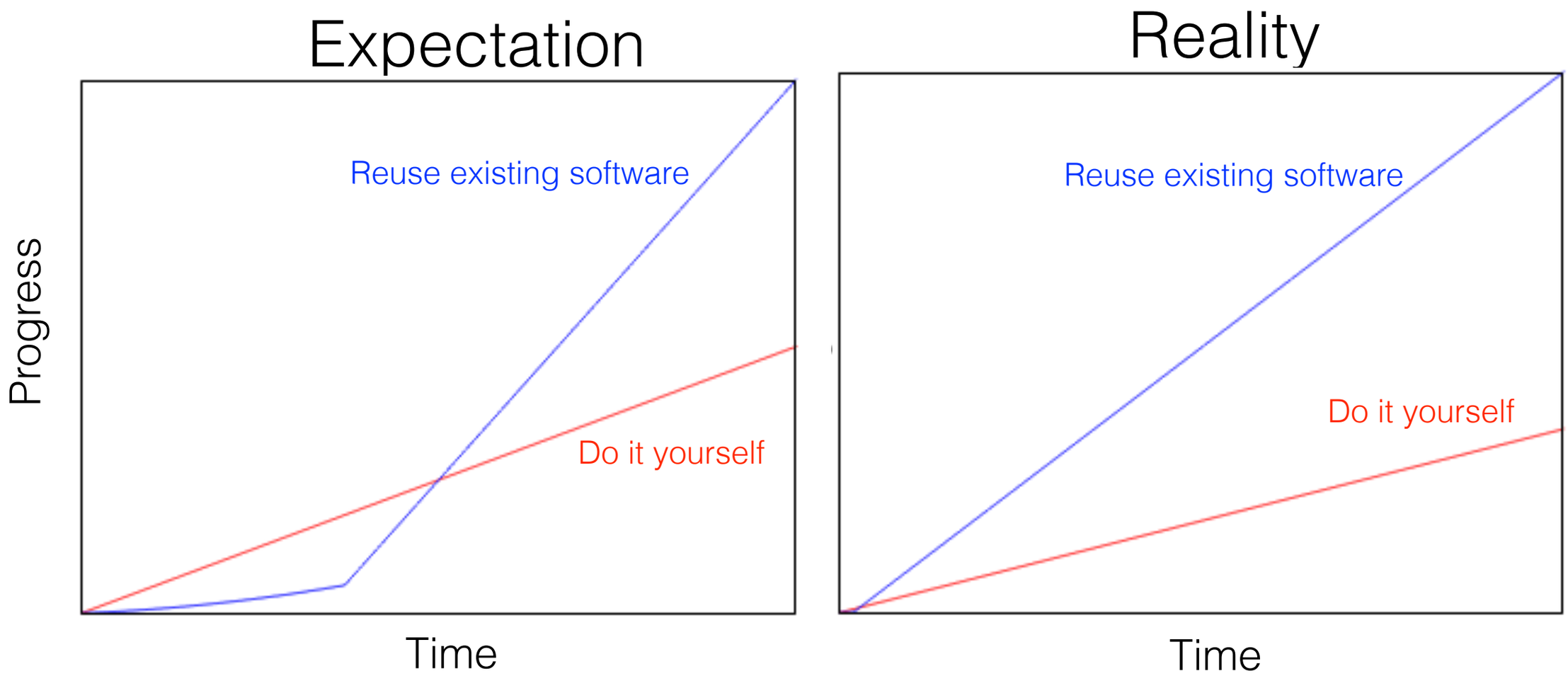
- deal.II user manual
  - The deal.II Library Documentation
  - Modules
  - Glossary
- deal.II tutorials and code gallery
  - Tutorial programs
  - The deal.II code gallery
- Us :-)
  - Don’t hesitate to ask questions

Introduction to deal.II

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

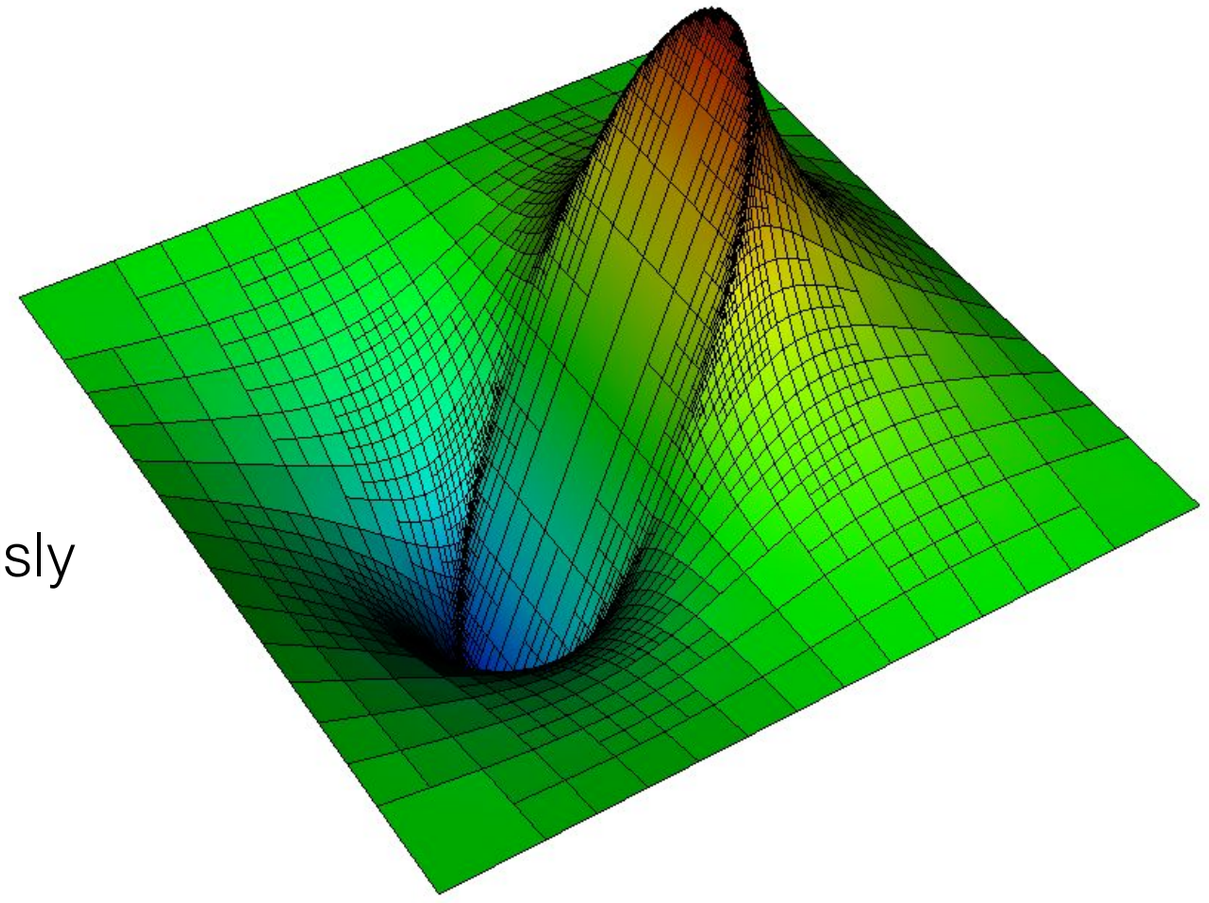
Why use deal.II (or any other PDE toolbox)?



- Applies to:
  - Users
  - Developers
- “The secret to good scientific software is (re)using existing libraries”

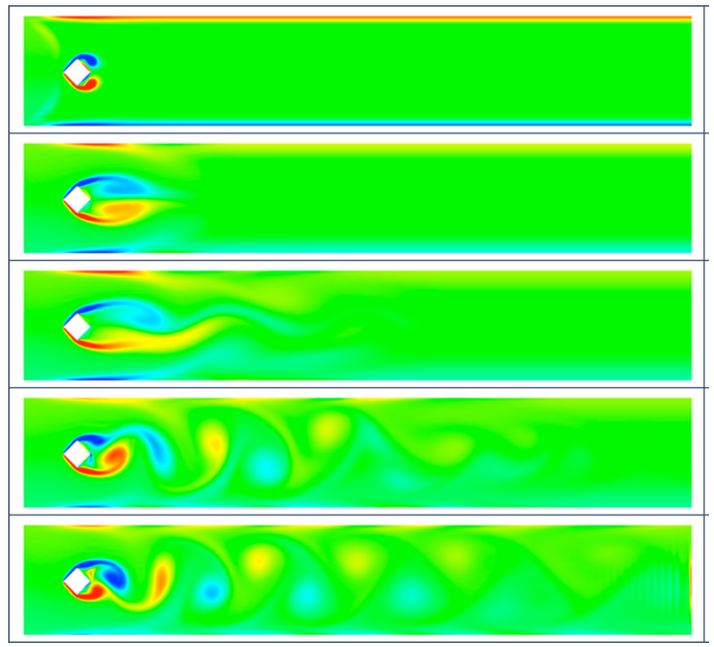
What is deal.II?
Differential Equation Analysis Library

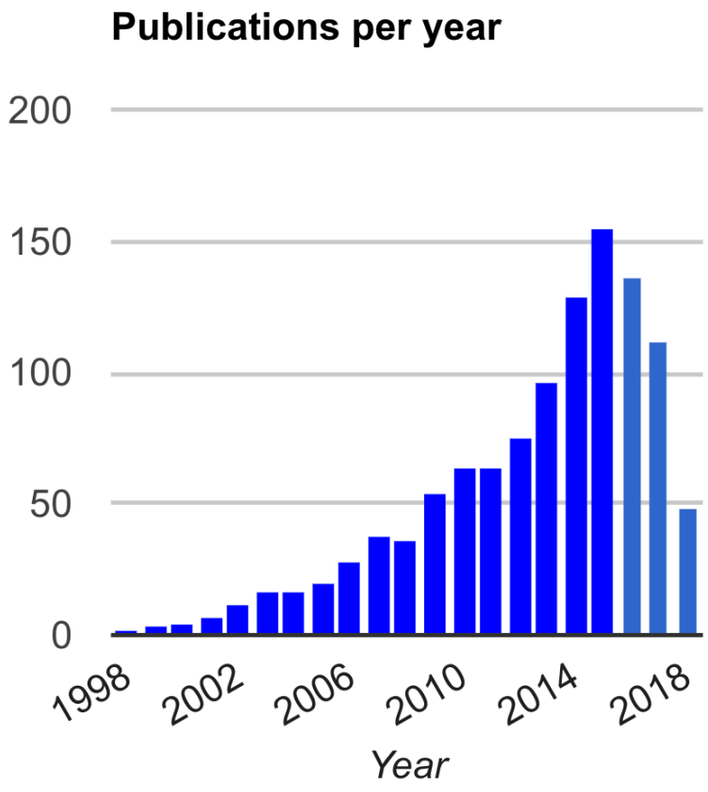
- Flexible open-source finite element toolkit
  - All the support functionality required to describe and solve a FE problem (PDEs)
  - Optimized for speed
  - Heavily tested
    - Many error checks (debug mode)
    - +10,000 regression tests run continuously
  - Part of SPEC CPU 2017 benchmark
- Templated C++ library (Object Orientated)
  - Dimension independent programming
- Portable
  - OS, architecture, compiler



What is deal.II?

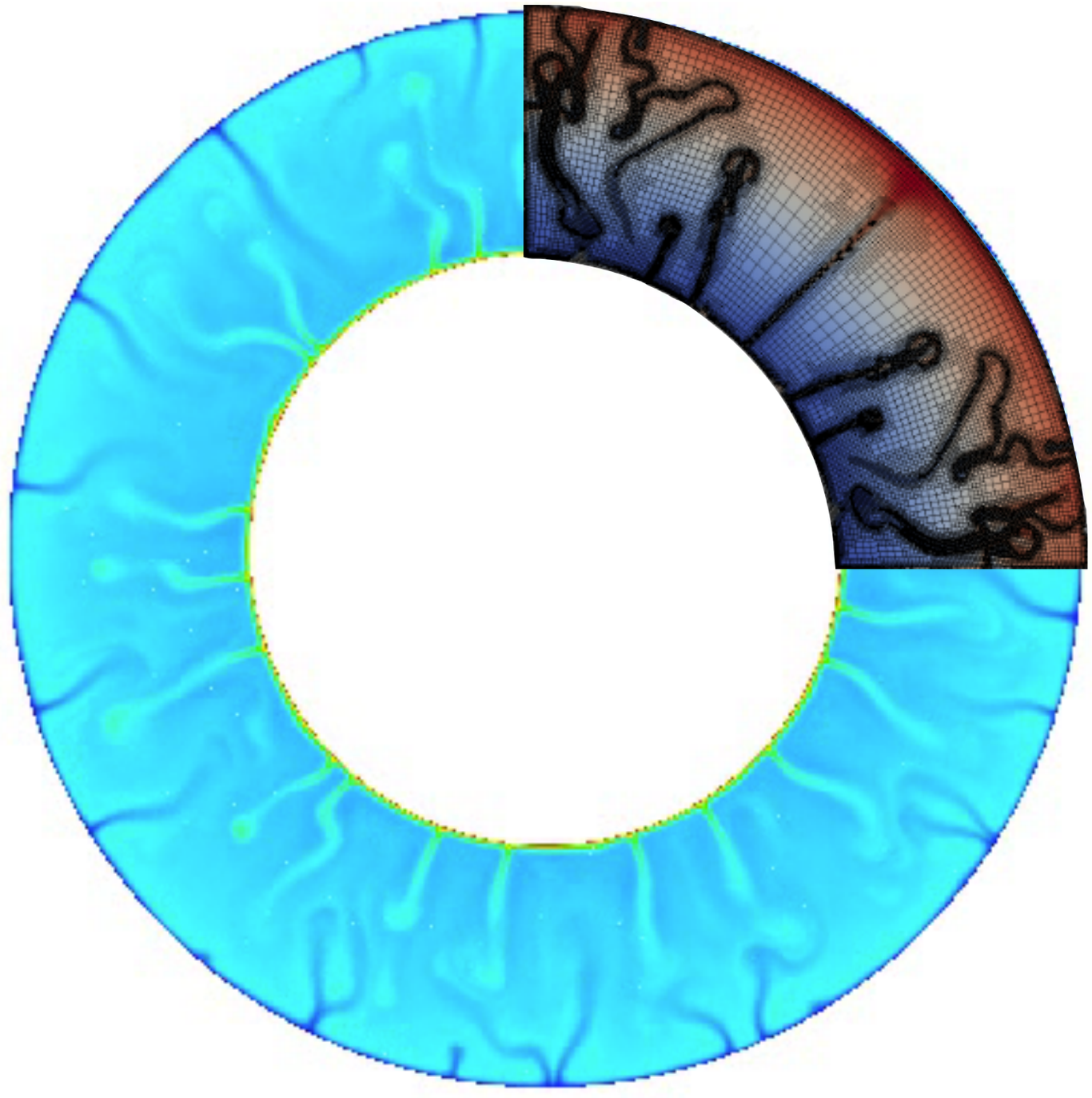
- Heavily documented
  - Over 10000 pages of interface documentation
  - Numerous tutorials
    - Illustrate functionality
    - Present methods to solve problems
- Quite widely used, and growing
- Active community
  - Approachable developers
  - Helpful online forum





Classes of problems solved using deal.II

- Geomechanics
- Fluid and gas dynamics
- Porous media
- Fluid-structure interaction
- Boundary element method
- Topology optimization
- Medical image reconstruction
- Structural mechanics
- Biomechanics
- Crystal growth
- Gradient and crystal plasticity
- Generalized continua
- Contact mechanics
- Atomistic-to-Continuum coupling
- Quantum mechanics
- Magneto- and electro-elasticity
- Thermo-plasticity

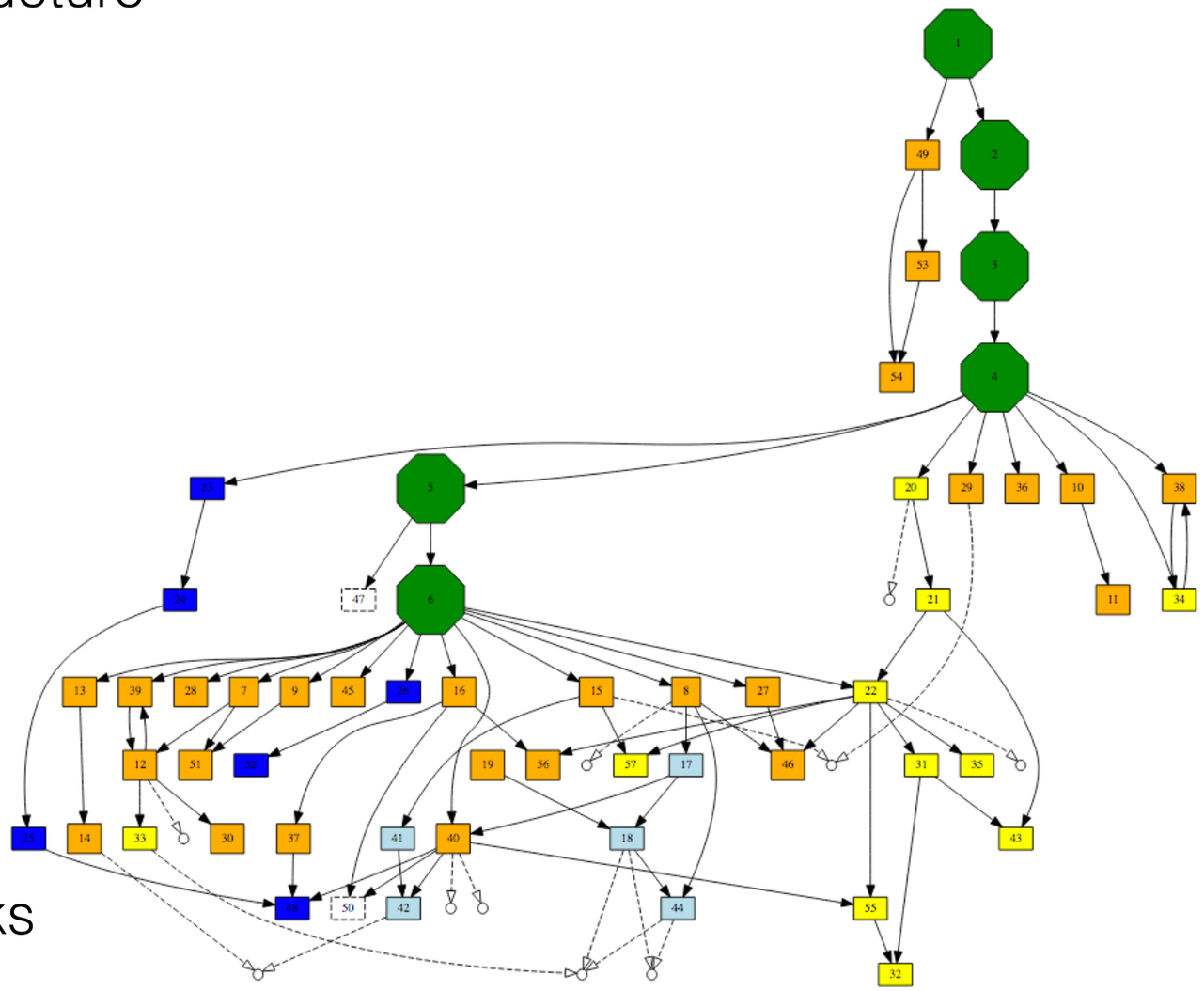


What deal.II is not

- A black box
  - Can’t throw any problem at it
  - Won’t do anything more than you ask it to
- Knows little* about
  - Numerical methods
  - Problem-specific details, i.e.
    - Preconditioners
    - Constitutive equations

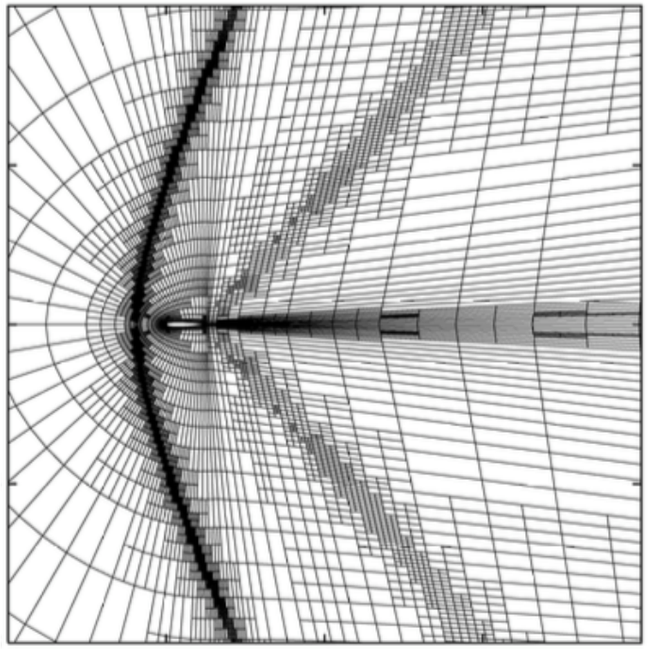
How deal.II will help you

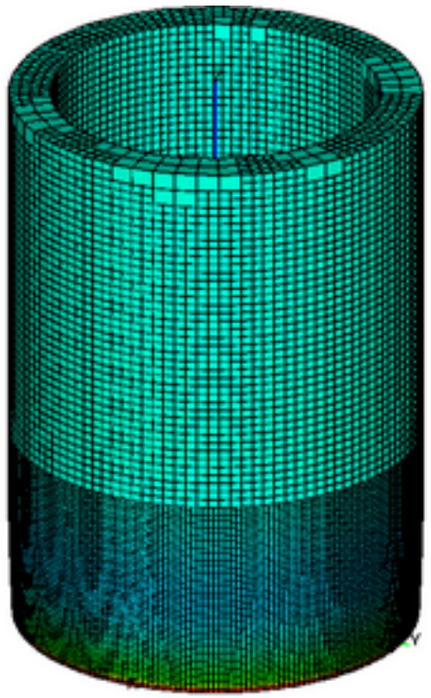
- Unified and well thought out data structure
  - Problem implementation
- Many tutorials
  - Baseline from which to build on
  - Demonstrate how to use features
- Comprehensive debugging support
  - Error messages everywhere!
- Some built in numerical tools
- Integration with advanced frameworks
  - Nonlinear solvers
  - Time integrators
  - Parallel sparse and dense linear algebra

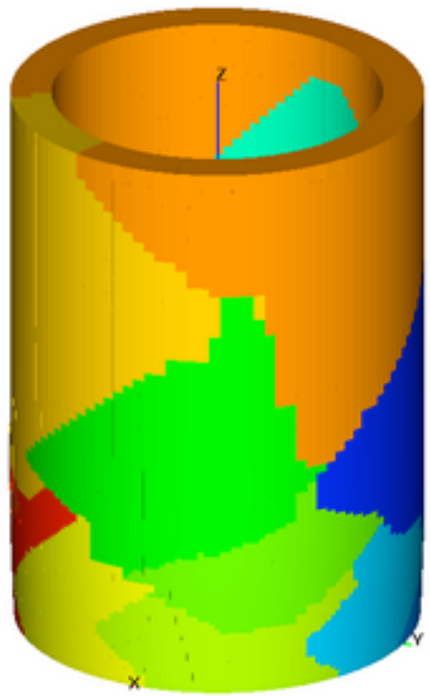


Fundamental capabilities and frameworks

- Mesh adaptivity
- Dense and sparse linear algebra
  - Built in tensor, dense matrix/vector classes
  - BLAS and LAPACK integration; GSL
  - Built in linear solvers and preconditioners
  - Eigenvalue solvers
- Parallelization
  - MPI
    - Linear algebra libraries (PETSc, Trilinos)
    - Distributed meshes ⟶ Billion DoFs
  - Threading (Intel TBB)
  - Vectorized numbers (AVX extensions)
- Pre/post-processing

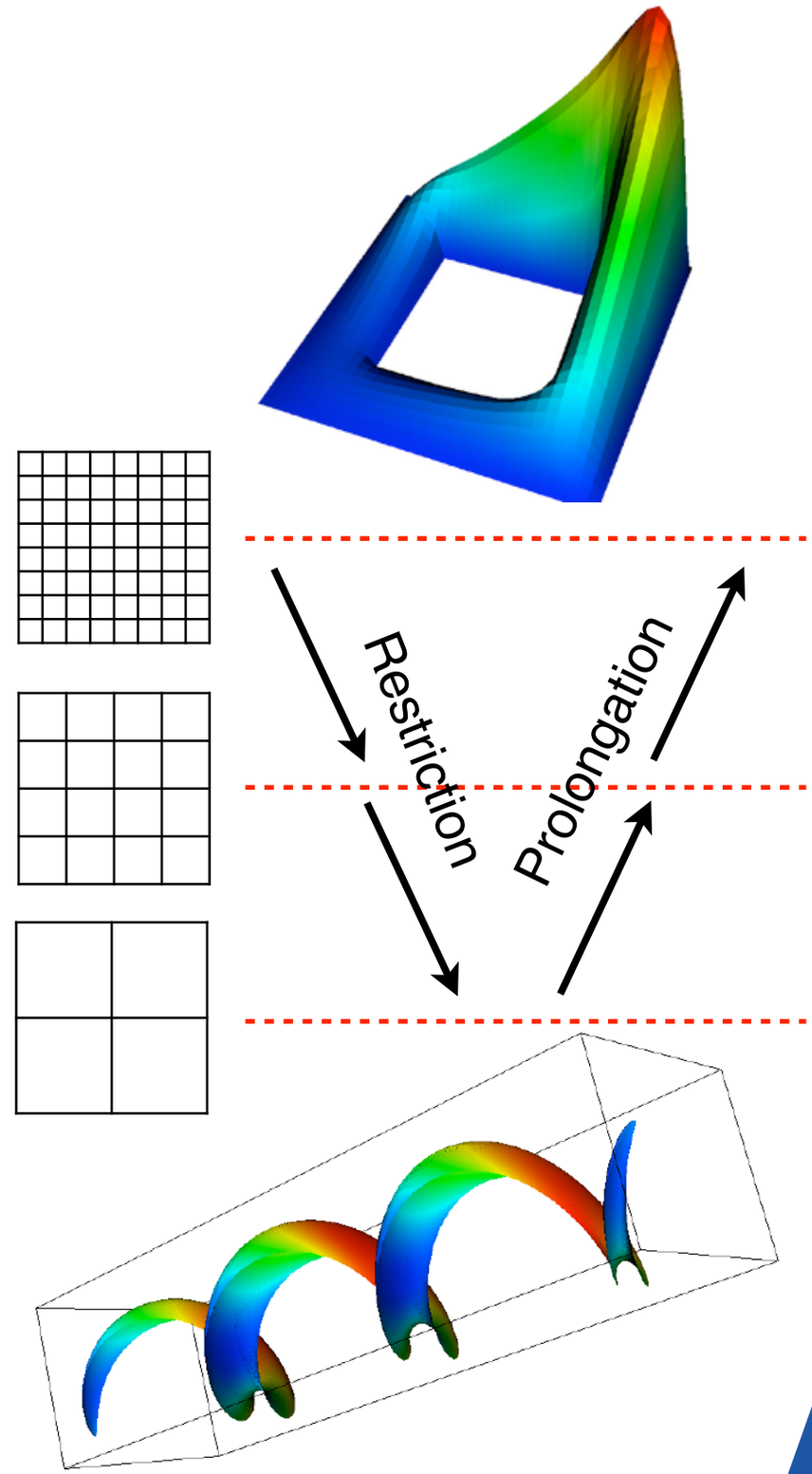






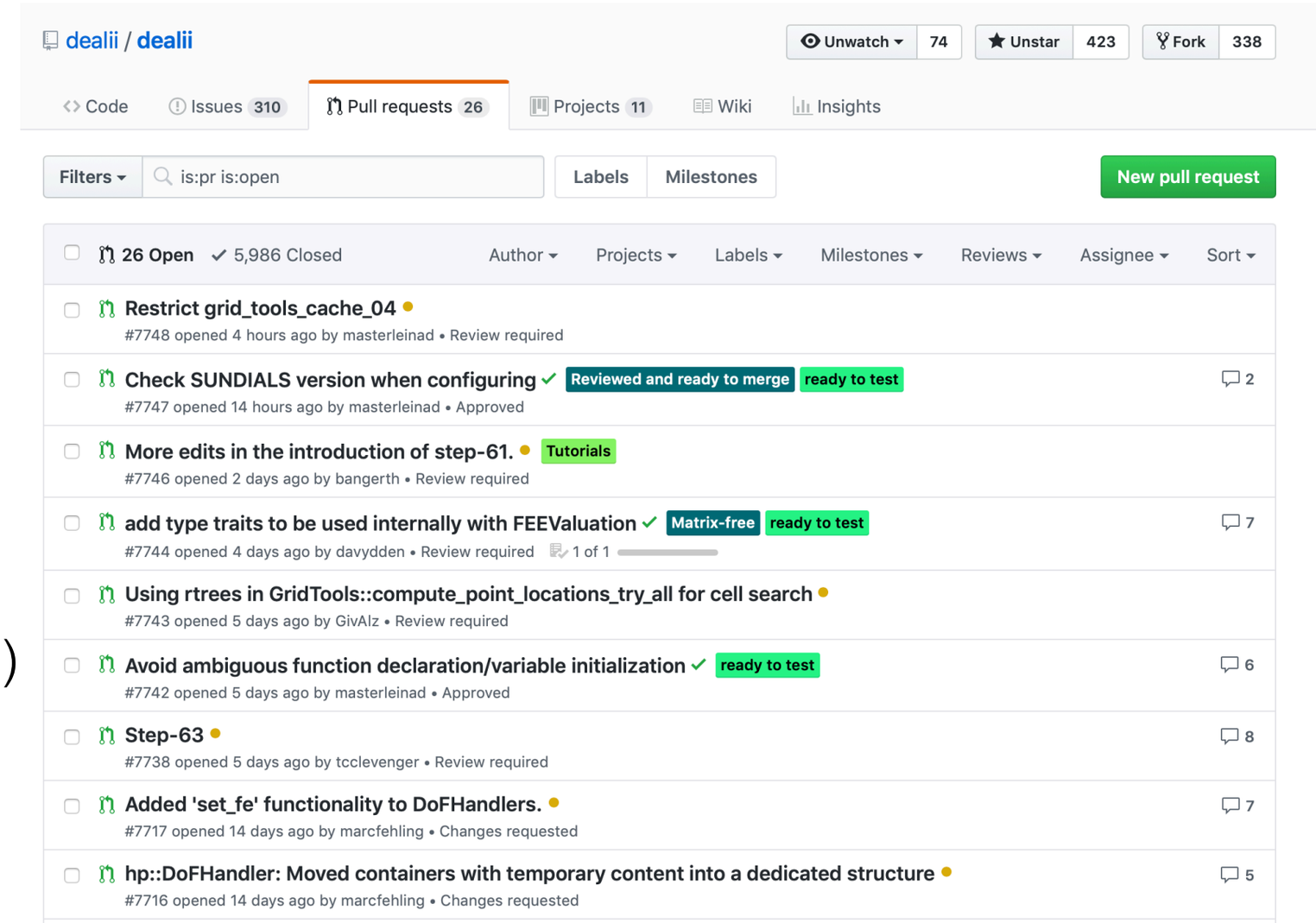
Advanced capabilities and frameworks

- hp-finite element support
- Meshworker
  - Assembly assistance
  - Functions to perform assembly for specific problem classes
- Geometric multi-grid
  - Using coarse grid as preconditioner to solution for finer grid
- Matrix-free
  - No explicit storing of matrix elements
  - Exchange memory transfer for computations
- Charts and manifolds
  - Accurate description of topologically complex objects



How deal.II is developed

- Open repository on GitHub
  - https://github.com/dealii/dealii
- Anyone can contribute!
- We encourage all to participate
- Every improvement is welcome (e.g. fix a typo in documentation)



Getting started with deal.II

Aims for this module

- Gain familiarity with two core classes
  - Triangulation
  - DoFHandler
- Create and interrogate meshes
- Create and interrogate sparsity patterns

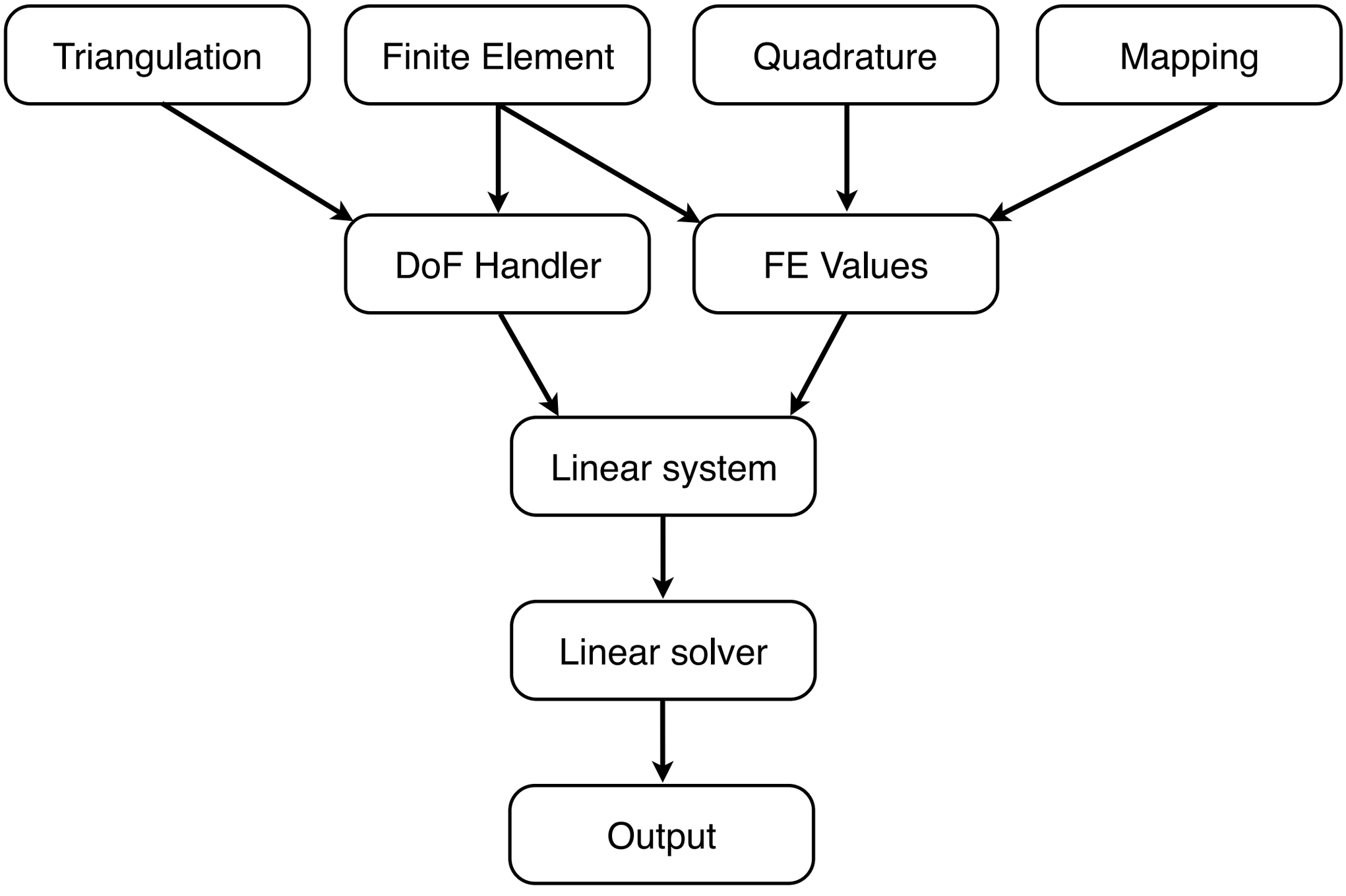
Reference material

- Main page
  The deal.II Library Documentation
- Tutorials
  - Step-1
  - Step-49
  - Step-2

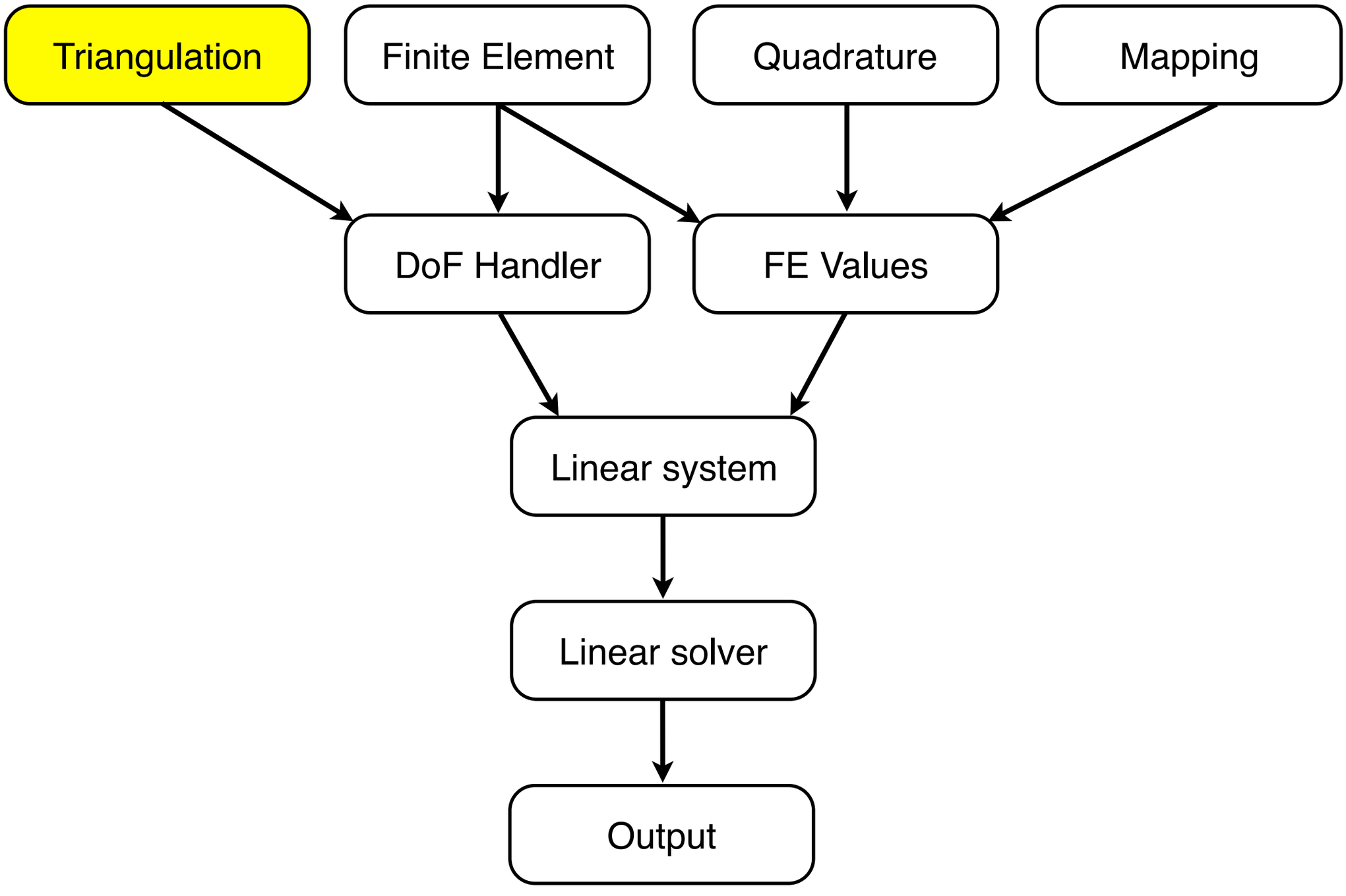
First and biggest tip

- Program defensively
  - Program and test in debug mode
    - Additional compiler warnings
    - Add assertions
  - Perform studies in release mode

Structure of a prototypical FE problem

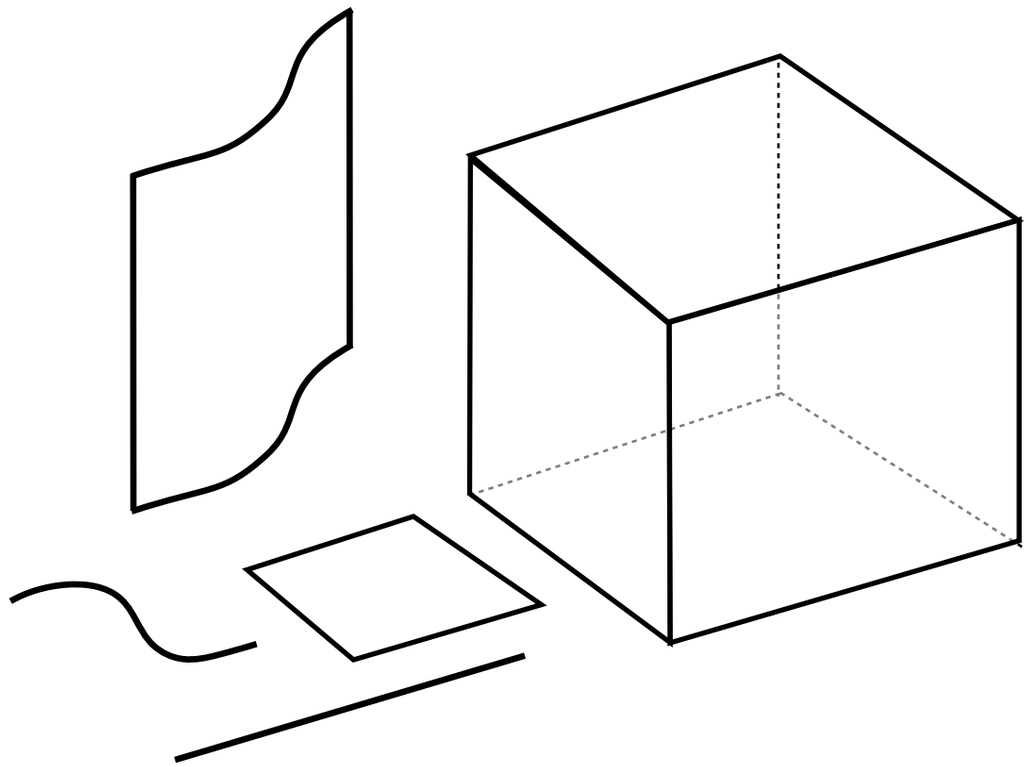


Structure of a prototypical FE problem



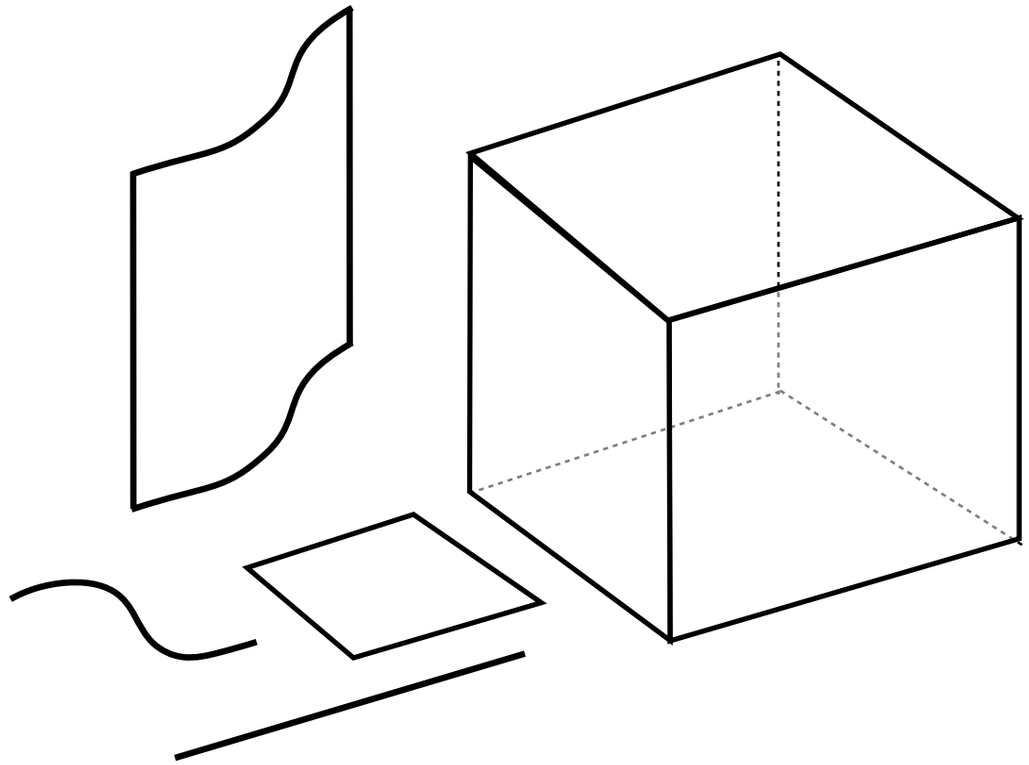
Interaction with geometry: the Triangulation class

- Describes problem geometry
  - Support for lines, quad, hex elements
  - Conceptually even higher order!
  - Structured/unstructured meshes
  - Co-dimension 1 or 2 case
- Grid creation
  - Built-in basic grid generation and manipulation tools (GridTools)
  - Can read in grids



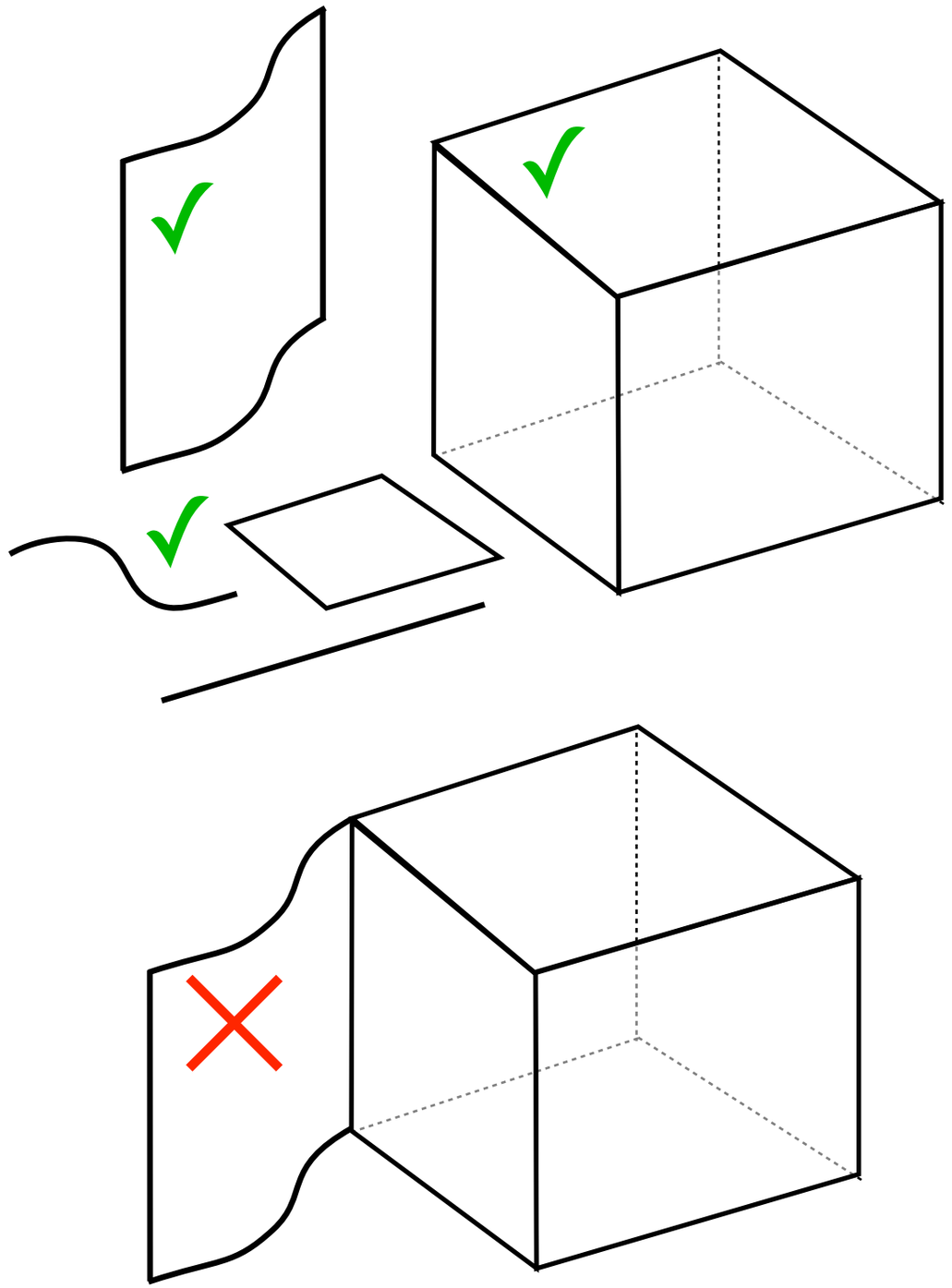
Interaction with geometry: the Triangulation class

- Assign helper ID's
  - Materials
  - Boundaries
  - Manifolds
- Allows storage of custom data-structure attached to each cell/face
- Cells know about neighbor cells
  - Useful for DG methods



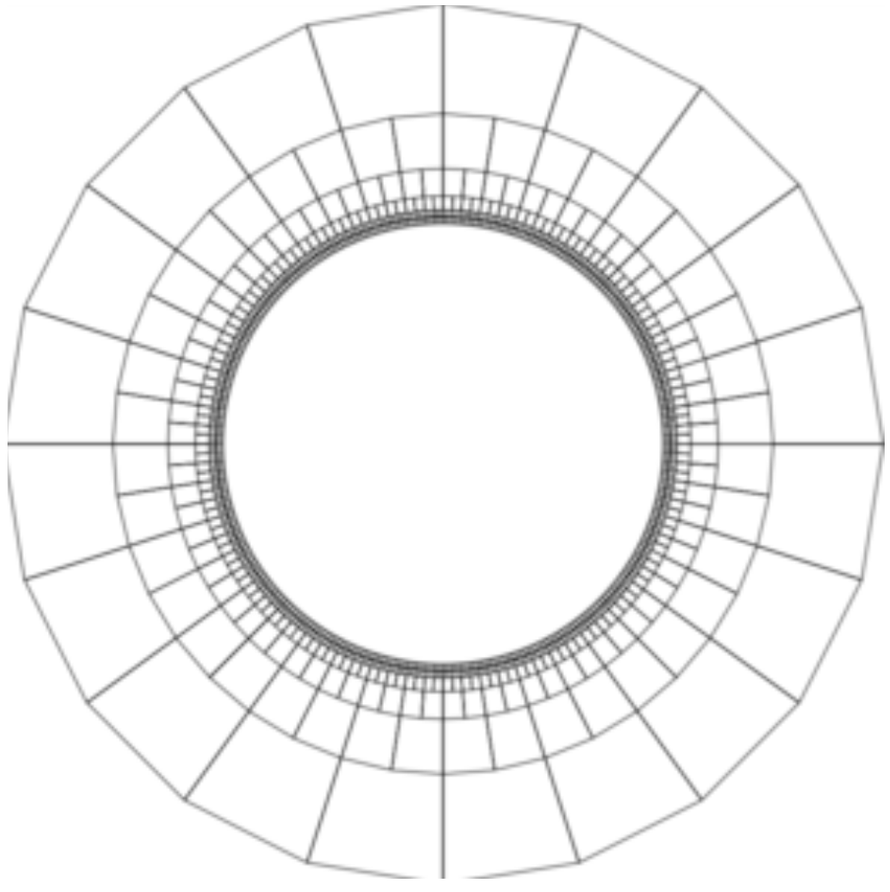
Interaction with geometry: the Triangulation class

- Can enforce topologies
  - Manifolds on boundary
  - Internal manifolds
- Disadvantage
  - Cannot mix triangulation types
  - e.g. Volumetric body with extended manifold surface

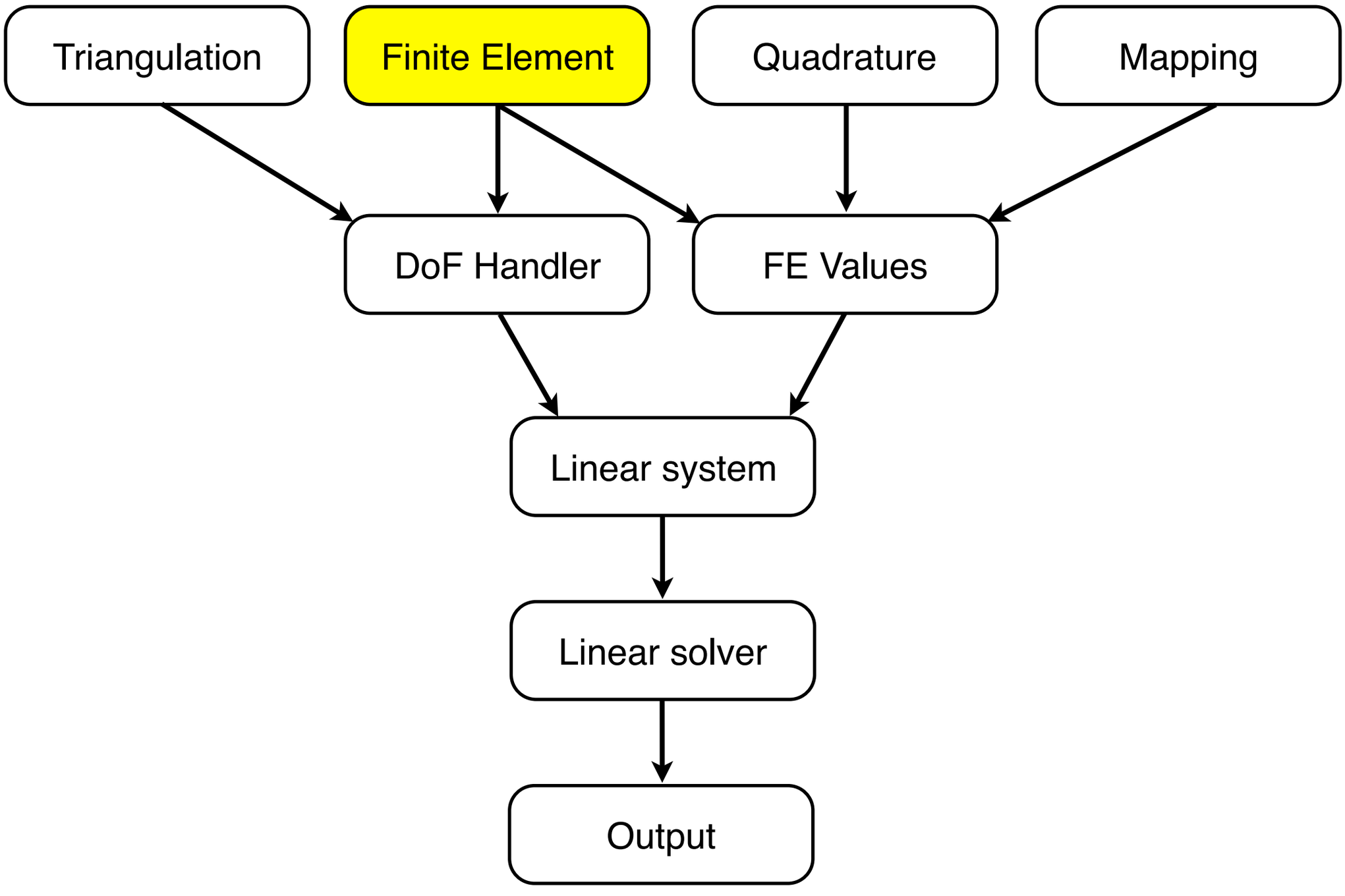


Interaction with geometry: the Triangulation class

- Demonstration: Step-1, step-49,
  Lecture 5: step-1, part 1: Simple meshes
  Lecture 6: step-1, part 2: Playing with meshes
- Key points
  - deal.II headers
  - Creating a triangulation
  - Boundary topology
  - Traversing a triangulation
  - Querying geometric information
  - Manipulating a triangulation
  - Aspects of grid refinement
  - Visualizing a triangulation



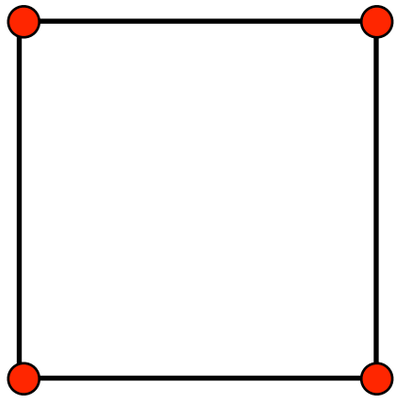
Structure of a prototypical FE problem



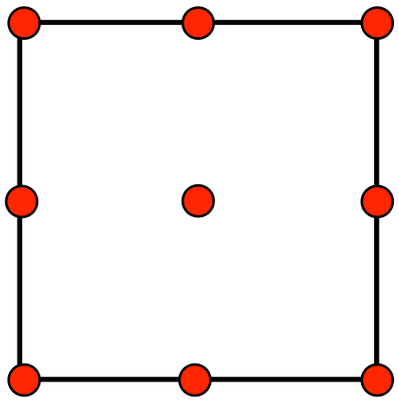
Assigning degrees-of-freedom: the FiniteElement classes

- Built in Finite Elements
  - Continuous
    - Piecewise Lagrange polynomials
  - Discontinuous
    - Monomials
    - Legendre polynomials
  - Vector-valued
    - Nedelec (H^{curl})
    - Raviart-Thomas (H^{div})
- Can develop finite elements from scratch
  - Specialization for FE's derived by polynomial expansions
  - Enhanced/bubble elements

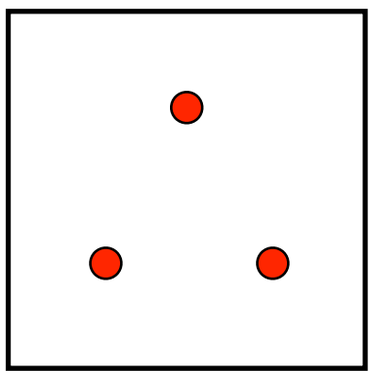
`FE_Q<2>(1)`



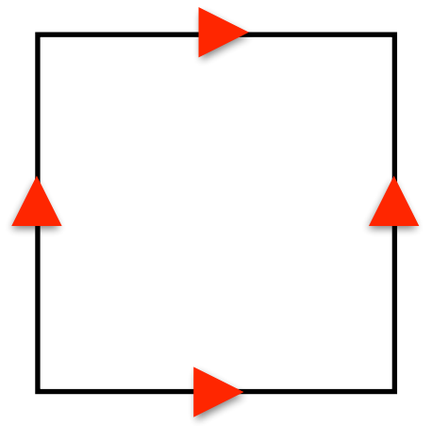
`FE_Q<2>(2)`



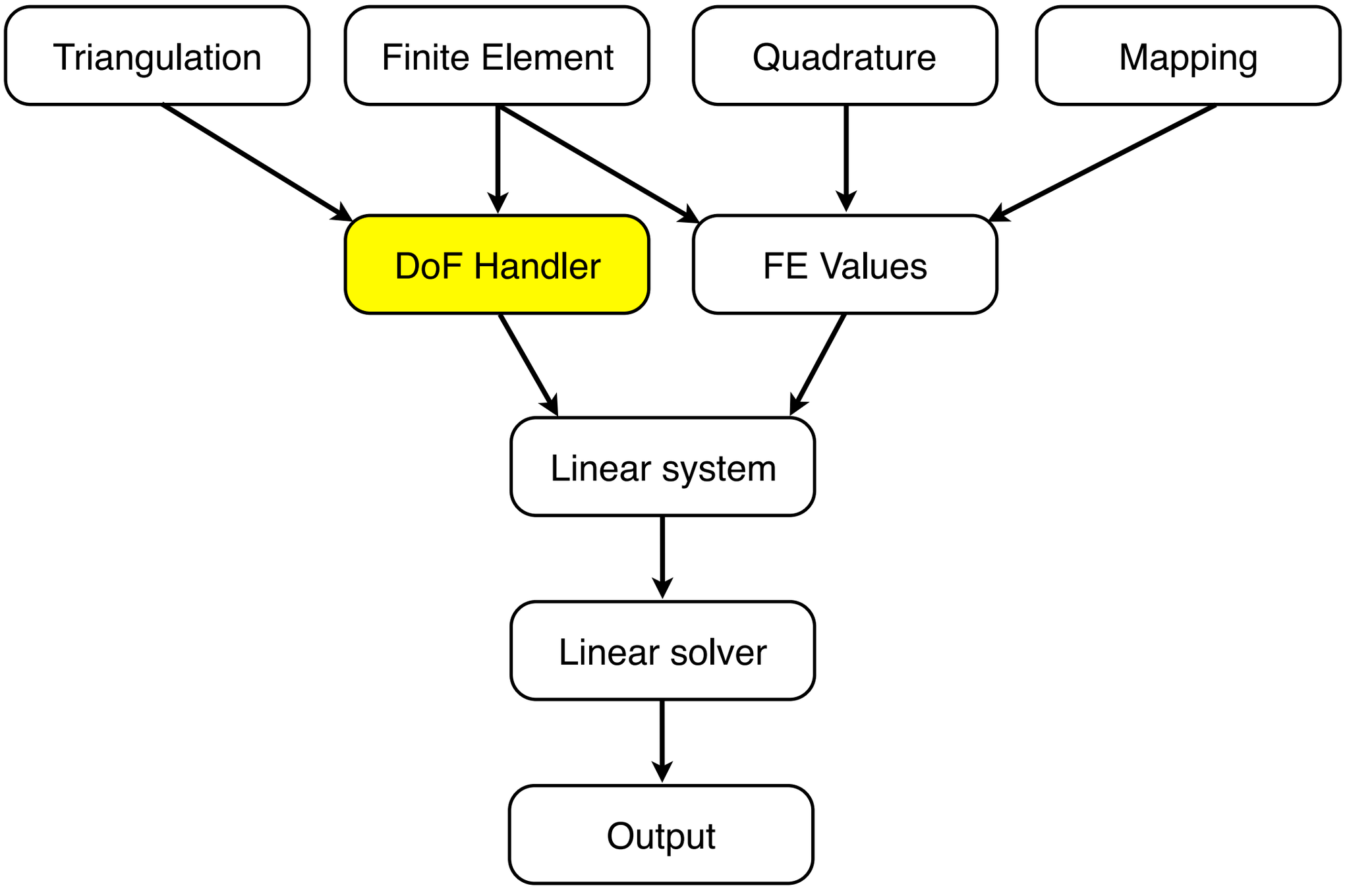
`FE_DGPMonomial<2>(1)`



`FE_Nedelec<2>(0)`

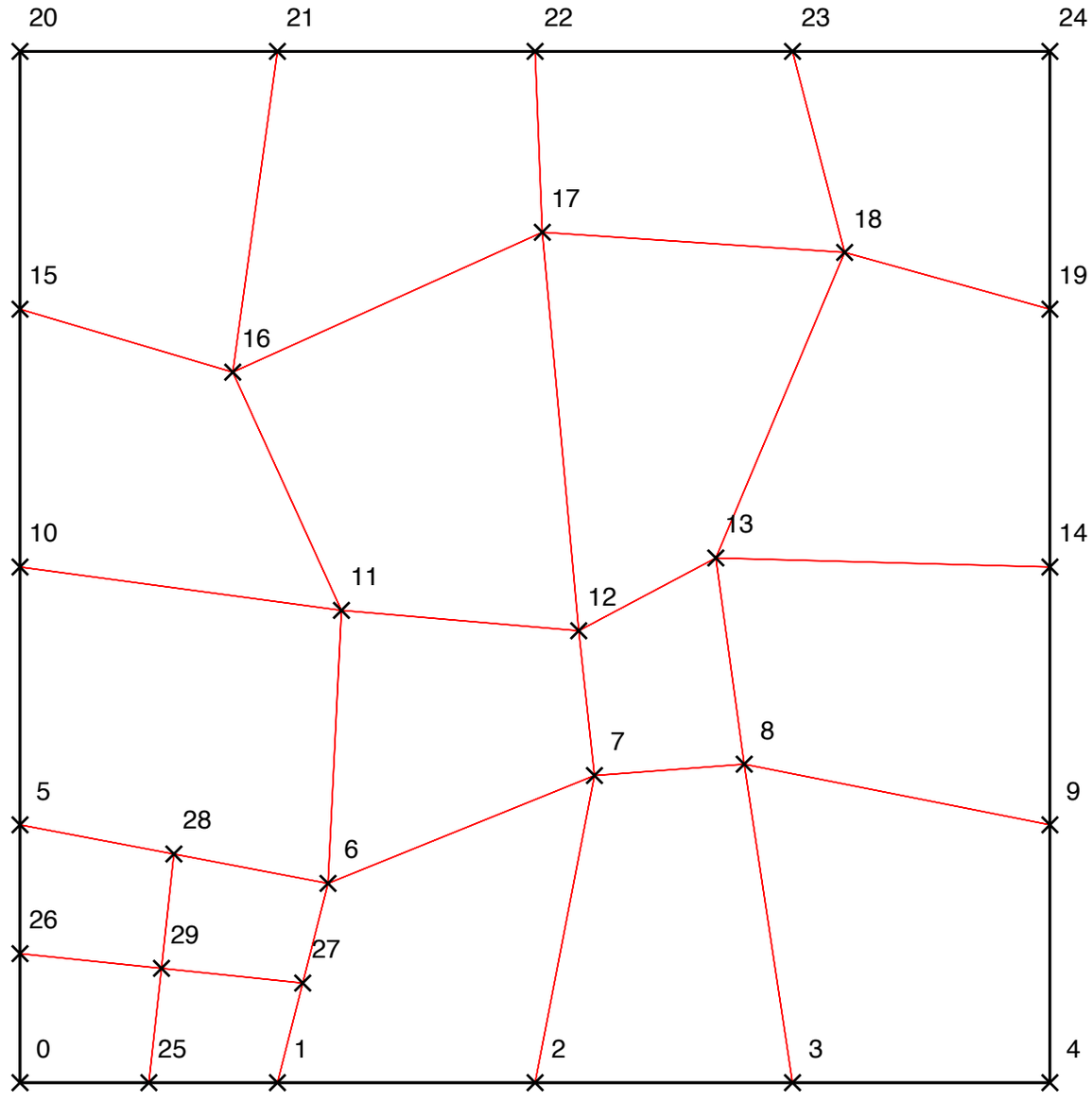


Structure of a prototypical FE problem



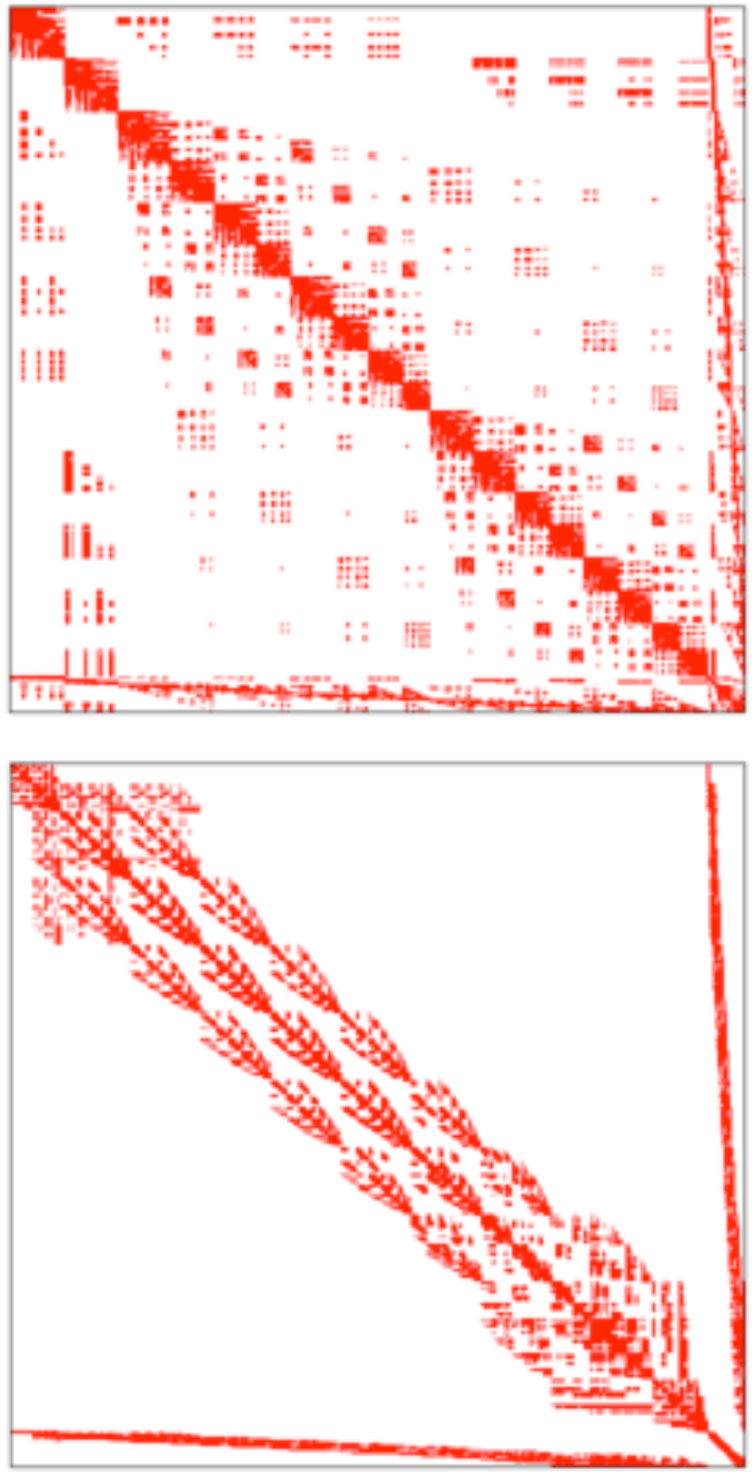
Assigning degrees-of-freedom: the DoFHandler class

- DoFHandler assigns DoF's to grid
  - Important: separate to Triangulation!
- Unified way to access DoF's, regardless of FE used
  - e.g. Discontinuous elements: support points not necessarily at vertices
- Fast access and grid traversal
  - STL-type cell iterators
  - Access to faces and edges from cells
- Disadvantage
  - Not straight-forward (but possible) to ask location of nodes



Assigning degrees-of-freedom: the DoFRenumbering namespace

- Renumbering schemes
  - Cuthill McKee
  - King
  - Downwind
- Reduce bandwidth (good for HPC and/or direct solvers)
- Collect like-components
- Induce block-structure
- Directional (fluid flow)
- MPI subdomain



Assigning degrees-of-freedom: the FiniteElement and DoFHandler classes

- Demonstration: Step-2
  http://www.math.colostate.edu/~bangerth/videos.676.9.html
- Key points
  - Choosing a Finite Element
  - Distributing degrees-of-freedom on a mesh
  - Renumbering degrees-of-freedom
  - Visualizing sparsity patterns