

deal.II Workshop @ hereon/TUHH

Day 1: Overview, FEM basics, mesh handling

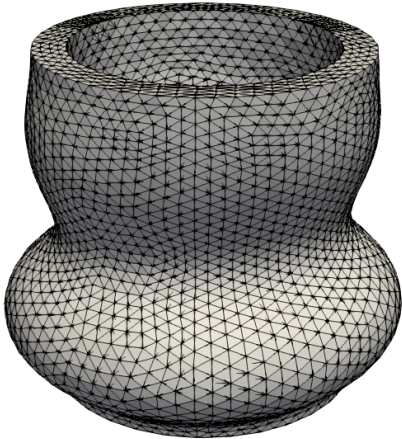
Peter Munch¹²³, Daniel Paukner³, Ingo Scheider³

¹deal.II developer

²Institute for Computational Mechanics, Technical University of Munich, Germany

³Kontinuumssimulationen, Institut für Werkstoffsystem-Modellierung, Helmholtz-Zentrum hereon, Germany

April 19, 2021



$$\text{Div}(\underline{\underline{F}} \cdot \underline{\underline{S}}(\underline{\underline{E}})) + \rho_0 \hat{\underline{\underline{b}}} = 0$$

FEM

How can deal.II help?

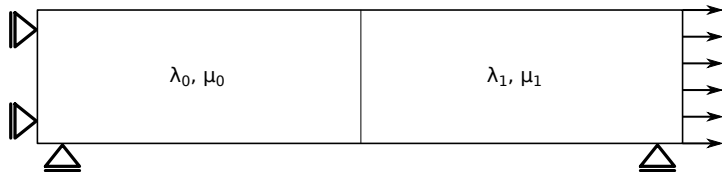
Organization:

- ▶ 9:00-12:00 (Monday-Wednesday): presentation/workshop with open end
- ▶ 9:00-9:30 (Monday): presentation round
- ▶ 9:00-9:30 (Tuesday, Wednesday): question time

Topics:

- ▶ Monday: introduction into FEM, overview of deal.II, mesh handling
- ▶ Tuesday: Poisson problem
- ▶ Wednesday: solid mechanics

Tension rod:

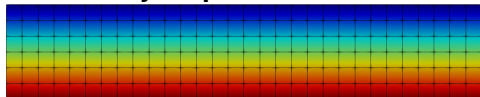


... with the result:

x displacement:



y displacement:



Part 1:

Introduction round

... other slide set

Part 2:

A short introduction into finite-element methods

Strong form of the Poisson problem:

$$\begin{aligned} -\nabla \cdot \nabla u &= f && \text{in } \Omega = (0, 1) \times (0, 1), \\ u &= h && \text{on } \Gamma_D = \{x = 0, y \in (0, 1)\}, \\ \nabla u(x, y) \cdot \underline{n} &= g && \text{on } \Gamma_N = \{x = 1, y \in (0, 1)\}, \\ \nabla u(x, y) \cdot \underline{n} &= 0 && \text{else.} \end{aligned}$$

Steps:

- definition of the function spaces
- derivation of the weak form
- spatial discretization + computation of the element stiffness matrix
- assembly and set-up of the linear equation system

- ▶ $\mathcal{V}_{\hat{u}}(t, \Omega) = \{u(\cdot, t) \in \mathcal{H}^1(\Omega) : u = \hat{u} \text{ on } \Gamma_D\}$ for the solution
- ▶ $\mathcal{V}_0(\Omega) = \{v \in \mathcal{H}^1(\Omega) : v = 0 \text{ on } \Gamma_D\}$ for the test function

1. step: multiplication with the test function v and integration over Ω

$$-\int_{\Omega} v(\underline{\mathbf{x}}) \nabla \cdot (\nabla u(\underline{\mathbf{x}}, t)) d\underline{\mathbf{x}} = \int_{\Omega} v(\underline{\mathbf{x}}) f d\underline{\mathbf{x}}$$

2. step: integration by parts:

$$\int_{\Omega} v(\underline{\mathbf{x}}) \nabla \cdot (\nabla u(\underline{\mathbf{x}})) d\underline{\mathbf{x}} = \int_{\Gamma} v(\underline{\mathbf{x}}) (\nabla u(\underline{\mathbf{x}})) \cdot \underline{\mathbf{n}} d\Gamma - \int_{\Omega} \nabla v(\underline{\mathbf{x}}) \cdot \nabla u(\underline{\mathbf{x}}) d\underline{\mathbf{x}}$$

3. step: exploitation of the boundary conditions (with: $\Gamma = \Gamma_D \cup \Gamma_N$):

$$\int_{\Gamma} v(\underline{\mathbf{x}}) \nabla u(\underline{\mathbf{x}}, t) \cdot \underline{\mathbf{n}} d\Gamma = \int_{\Gamma_D} v(\underline{\mathbf{x}}) \nabla u(\underline{\mathbf{x}}, t) \cdot \underline{\mathbf{n}} d\Gamma + \overset{0}{\int_{\Gamma_N} v(\underline{\mathbf{x}}) g d\Gamma}$$

Weak form

Find $u(\underline{\mathbf{x}}) \in \mathcal{V}_{\hat{u}}$ such that for all $v(\underline{\mathbf{x}}) \in \mathcal{V}_0(\Omega)$:

$$\int_{\Omega} \nabla v(\underline{\mathbf{x}}) \cdot \nabla u(\underline{\mathbf{x}}) d\underline{\mathbf{x}} = \int_{\Omega} v(\underline{\mathbf{x}}) f d\underline{\mathbf{x}} + \int_{\Gamma_N} v(\underline{\mathbf{x}}) g d\Gamma$$

in compact notation:

$$(\nabla v(\underline{\mathbf{x}}), \nabla u(\underline{\mathbf{x}}))_{\Omega} = (v(\underline{\mathbf{x}}), f)_{\Omega} + (v(\underline{\mathbf{x}}), g)_{\Gamma_N}$$

- ▶ decompose computational domain into cells $\Omega = \bigcup \Omega^{(e)}$
- ▶ use scalar Lagrange finite element \mathcal{Q}_k :

$$(\nabla v, \nabla u)_{\Omega_e} \approx \sum_q (\nabla v, \nabla u) \cdot |J| \times w \quad \rightarrow \quad \mathbf{K}_{ij}^{(e)} = \sum_q (\nabla N_{iq}, \nabla N_{jq}) \cdot |J_q| \times w_q$$

$$(v, f)_{\Omega_e} \quad \rightarrow \quad \mathbf{f}_i^{(e)} = \sum_q (N_{iq}, f) \cdot |J_q| \times w_q$$

$$(v, g)_{\Gamma_{e,N}} \quad \rightarrow \quad \mathbf{g}_i^{(e)} = \sum_q (N_{iq}, g) \cdot |J_q| \times w_q$$

... with N shape functions in real space, mapping & quadrature

- ▶ loop over all cells, assemble system matrix and right-hand-side vector, and solve system

$$\mathbf{K} \mathbf{u} = \mathbf{f} + \mathbf{g}$$

The solution of a PDE with an FEM library (like deal.II):

$$\mathbf{K}\mathbf{u} = \mathbf{f} + \mathbf{g} \quad \text{with} \quad \mathbf{K}_{ij}^{(e)} = \sum_q (\nabla N_{iq}, \nabla N_{jq}) \cdot |J_q| \times w_q$$

$$\mathbf{f}_i^{(e)} = \sum_q (N_{iq}, f) \cdot |J_q| \times w_q$$

$$\mathbf{g}_i^{(e)} = \sum_q (N_{iq}, g) \cdot |J_q| \times w_q$$

requires:

- ▶ mesh handling
- ▶ finite elements, quadrature rules, mapping rules
- ▶ assembly procedure
- ▶ linear solver

Part 3:

Short overview of deal.II

- ▶ deal.II¹: mathematical software for finite-element analysis, written in C++
- ▶ origin in Heidelberg 1998: Wolfgang Bangerth, Ralf Hartmann, Guido Kanschat
- ▶ 275 contributors + principal developer team with 11 active members
- ▶ more than 1,600 publications (on and with deal.II)
- ▶ freely available under LGPL 2.1 license
- ▶ yearly releases; current release: 9.2
- ▶ features comprise: matrix-free implementations, parallelization (MPI, threading via TBB & Taskflow, SIMD, GPU support), discontinuous Galerkin methods, AMR via p4est, particles, wrappers for PETSc and Trilinos, ...

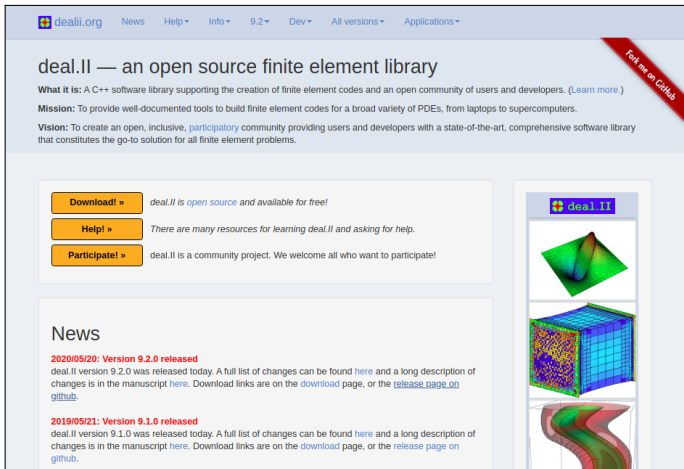


¹ successor of DEAL: Differential Equations Analysis Library

Publications describing the design of and recent development in deal.II:

D. Arndt, W. Bangerth, D. Davydov, T. Heister, L. Heltai, M. Kronbichler, M. Maier, J.-P. Pelteret, B. Turcksin, and D. Wells. [The deal.II finite element library: Design, features, and insights](#). *Computers and Mathematics with Applications*. 2020. DOI: <https://doi.org/10.1016/j.camwa.2020.02.022>

D. Arndt, W. Bangerth, B. Blais, T. C. Clevenger, M. Fehling, A. V. Grayver, T. Heister, L. Heltai, M. Kronbichler, M. Maier, P. Munch, J.-P. Pelteret, R. Rastak, I. Thomas, B. Turcksin, Z. Wang, and D. Wells. [The deal.II Library, Version 9.2](#). *Journal of Numerical Mathematics*. 2020. DOI: <https://doi.org/10.1515/jnma-2020-0043>



The screenshot shows the official dealii.org website. At the top is a navigation bar with links for News, Help, Info, 9.2, Dev, All versions, and Applications. The main heading is "deal.II — an open source finite element library". Below this, it states "What it is: A C++ software library supporting the creation of finite element codes and an open community of users and developers." It also lists the "Mission" and "Vision" of the project. A red banner on the right says "Fork me on GitHub". In the center, there are three buttons: "Download!", "Help!", and "Participate!", each with a brief description. To the right of these buttons is a vertical stack of three images: a 3D surface plot, a 2D mesh of a square, and a 3D surface plot of a curved surface. Below the buttons is a "News" section with two entries: "2020/05/20: Version 9.2.0 released" and "2019/05/21: Version 9.1.0 released", each with a description of the release and links to the full list of changes, download page, and release page on GitHub.

dealii.org News Help Info 9.2 Dev All versions Applications

deal.II — an open source finite element library

What it is: A C++ software library supporting the creation of finite element codes and an open community of users and developers. ([Learn more.](#))

Mission: To provide well-documented tools to build finite element codes for a broad variety of PDEs, from laptops to supercomputers.

Vision: To create an open, inclusive, [participatory](#) community providing users and developers with a state-of-the-art, comprehensive software library that constitutes the go-to solution for all finite element problems.

Download! » *deal.II is [open source](#) and available for free!*

Help! » *There are many resources for learning deal.II and asking for help.*


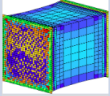

Participate! » *deal.II is a community project. We welcome all who want to participate!*

News

2020/05/20: Version 9.2.0 released
deal.II version 9.2.0 was released today. A full list of changes can be found [here](#) and a long description of changes is in the manuscript [here](#). Download links are on the [download](#) page, or the [release page on github](#).

2019/05/21: Version 9.1.0 released
deal.II version 9.1.0 was released today. A full list of changes can be found [here](#) and a long description of changes is in the manuscript [here](#). Download links are on the [download](#) page, or the [release page on github](#).

deal.II



... www.dealii.org

deal.II Reference documentation for deal.II version Git 899b683bb8 2020-11-07 12:42:48 -0500

Main Page Tutorial Code gallery Modules Namespaces Classes Related Pages Files dealii.org

VectorizedArray< Number, width > Class Template Reference

`#include <deal.II/base/vectorization.h>`

Inheritance diagram for VectorizedArray< Number, width >:

```

graph TD
    VAB1["VectorizedArrayBase< T, width >"]
    VAB2["VectorizedArrayBase< VectorizedArray< Number, width >, 1 >"]
    VA["VectorizedArray< Number, width >"]
    D["< double >"]
    N["< Number >"]
    VAD["VectorizedArray< double >"]
    VAN["VectorizedArray< Number >"]

    VAB1 -- "< VectorizedArray< Number, width >, 1 >" --> VAB2
    VAB2 --> VA
    VA -- "< double >" --> VAD
    VA -- "< Number >" --> VAN
    
```

Public Types

using `value_type` = Number

Public Member Functions

`VectorizedArray` ()=default
`VectorizedArray` (const Number scalar)

Extensive Doxygen documentation

dealii / dealii

Unwatch 62 Star 678 Fork 478

Code Issues 462 Pull requests 55 Actions Projects 19 Wiki Security Insights

Frequently Asked Questions

Jens edited this page on Sep 7 · 52 revisions

The deal.II FAQ

This page collects a few answers to questions that have frequently been asked about deal.II and that we thought are worth recording as they may be useful to others as well.

Table of Contents

- The deal.II FAQ
 - Table of Contents
 - General questions on deal.II
 - Can I use/implement triangles/tetrahedra in deal.II?
 - I'm stuck!
 - I'm not sure the mailing list is the right place to ask ...
 - How fast is deal.II?
 - deal.II programs behave differently in 1d than in 2/3d
 - I want to use deal.II for work in my company. Do I need a special license?
 - Supported System Architectures
 - Can I use deal.II on a Windows platform?
 - Run deal.II in the Windows Subsystem for Linux
 - Run deal.II natively on Windows
 - Run deal.II through a virtual box

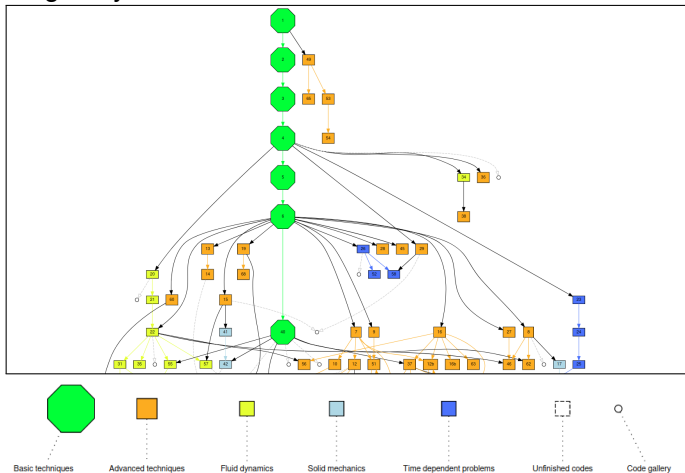
Pages 629

Find a Page...

- Home
- Code DOI best practices
- Contributing
- deal.II in Spack
- deal.II on Homebrew Linuxbrew
- Debugging with GDB
- Docker Images
- DoF Handler
- Eclipse
- Electromagnetic problem
- Emacs
- Frequently Asked Questions
- Function Plotting Tool
- Gallery

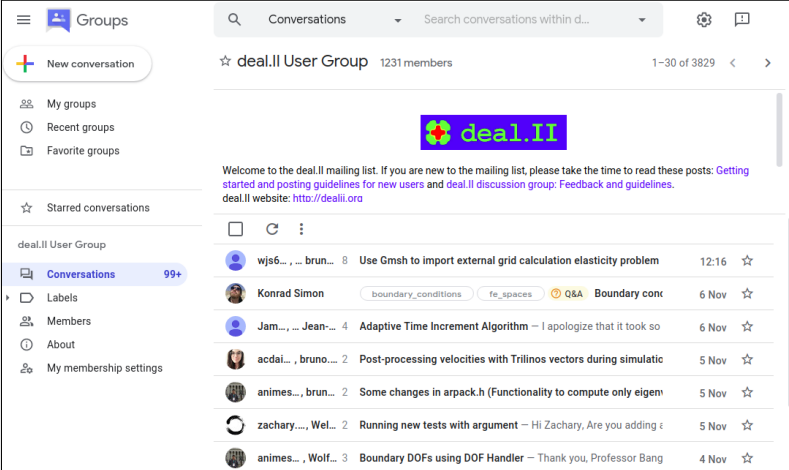
GitHub Wiki

70 tutorials and code gallery:










... further 7 tutorials: work in progress

deal.II user group:



The screenshot shows the deal.II User Group forum page. The left sidebar contains navigation options: 'New conversation', 'My groups', 'Recent groups', 'Favorite groups', 'Starred conversations', and a list of group tabs including 'Conversations' (99+), 'Labels', 'Members', 'About', and 'My membership settings'. The main content area displays the group name 'deal.II User Group' with 1231 members and a search bar. Below the group name is a welcome message and a list of recent discussions. The discussions are as follows:

Avatar	Author	Topic	Time	Star
	wjs6..., ... brun...	Use Gmsh to import external grid calculation elasticity problem	12:16	☆
	Konrad Simon	boundary_conditions fe_spaces Q&A Boundary conc	6 Nov	☆
	Jam..., ... Jean...	Adaptive Time Increment Algorithm — I apologize that it took so	6 Nov	☆
	acda..., bruno...	Post-processing velocities with Trilinos vectors during simulatio	5 Nov	☆
	animes..., brun...	Some changes in arpack.h (Functionality to compute only eigen	5 Nov	☆
	zachary..., Wel...	Running new tests with argument — Hi Zachary, Are you adding ε	5 Nov	☆
	animes..., Wolf...	Boundary DOFs using DOF Handler — Thank you, Professor Bang	4 Nov	☆

... Q&A by users and developers!

- ▶ issues
- ▶ pull requests
- ▶ GitHub actions → CI
- ▶ required: approval by ≥ 1 principal developer

Login All Dashboards Saturday, November 07 2020 18:48:28

Deal.II

Dashboard Calendar Previous Current Project

3 hours ago: 13 tests failed on GNU-9.3.0-master-debian-11
4 hours ago: 12 tests failed on GNU-9.3.0-master-ubuntu-lts-20
7 hours ago: 28 tests failed on GNU-9.3.0-master
7 hours ago: 1 warning introduced on GNU-9.3.0-master
8 hours ago: 38 tests failed on GNU-10.1.0-master-tets

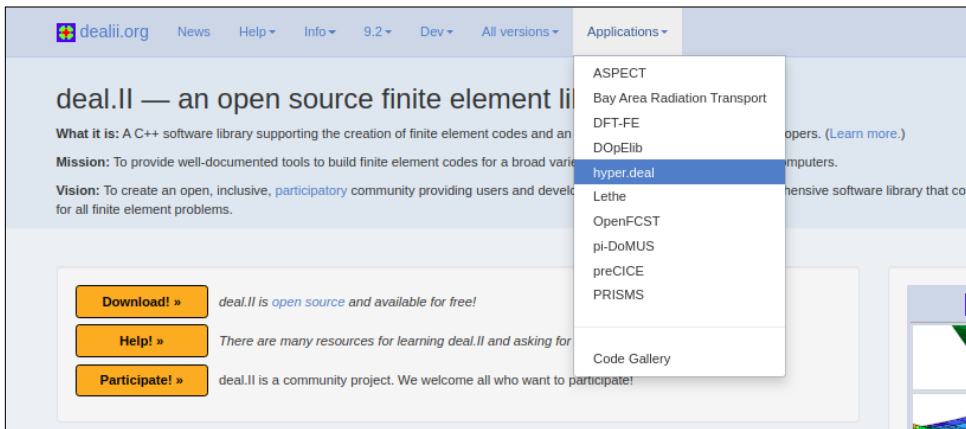
[See full feed](#)

Regression Tests 150 builds

Site	Build Name	Update	Configure		Build		Test			Start Time
		Revision	Error	Warn	Error	Warn	Not Run	Fall	Pass	
tester	Clang-10.0.0-master-avx2-Ofast	10ceee	0	0	0	0	0	50 ⁺¹	12417 ⁺⁶	20 hours ago
tester	Clang-10.0.0-master-avx2-Ofast	414559	0	0	0	0	0	49 ⁺¹	12404 ⁻¹	Nov 03, 2020 - 07:54 UTC
tester	Clang-10.0.0-master-avx2-Ofast	2bdb45	0	0	0	0	0	48	12417 ⁺²	Nov 06, 2020 - 01:00 UTC
tester	Clang-10.0.0-master-avx2-Ofast	cd2846	0	0	0	0	0	48	12415 ⁺⁶	Nov 05, 2020 - 03:17 UTC
tester	Clang-10.0.0-master-avx2-Ofast	220f05	0	0	0	0	0	48 ⁻¹	12415 ⁺¹¹	Nov 04, 2020 - 05:34 UTC
tester	GNU-10.1.0-master-tets	10ceee	0	0	0	0	0	38	11866 ⁺²	10 hours ago
tester	GNU-10.1.0-master-tets	2bdb45	0	0	0	0	0	38	11864 ⁺²	Nov 06, 2020 - 10:56 UTC

... more than 5,000 tests run for different compilers/hardware/configurations

Some deal.II-based user codes/libraries are open source as well:



The screenshot shows the dealii.org website. The navigation bar includes links for News, Help, Info, 9.2, Dev, All versions, and Applications. The Applications dropdown menu is open, displaying a list of user codes/libraries: ASPECT, Bay Area Radiation Transport, DFT-FE, DOpElib, hyper.deal (highlighted), Lethe, OpenFCST, pi-DoMUS, preCICE, and PRISMS. Below the menu, there is a 'Code Gallery' link. The main content area features the heading 'deal.II — an open source finite element library' and a description of the library's purpose. Below this, there are three orange buttons: 'Download! »', 'Help! »', and 'Participate! »', each followed by a short description of the deal.II project.

dealii.org News Help Info 9.2 Dev All versions Applications

deal.II — an open source finite element library

What it is: A C++ software library supporting the creation of finite element codes and an efficient solver for a broad variety of problems. (Learn more.)

Mission: To provide well-documented tools to build finite element codes for a broad variety of problems on computers.

Vision: To create an open, inclusive, participatory community providing users and developers with a comprehensive software library that covers all finite element problems.

Download! » deal.II is open source and available for free!

Help! » There are many resources for learning deal.II and asking for help.

Participate! » deal.II is a community project. We welcome all who want to participate!

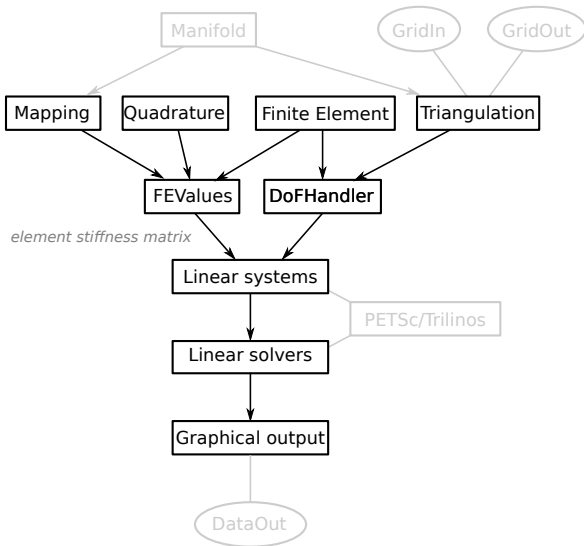
ASPECT
Bay Area Radiation Transport
DFT-FE
DOpElib
hyper.deal
Lethe
OpenFCST
pi-DoMUS
preCICE
PRISMS
Code Gallery

... motivation for further development

needed from a FEM library:

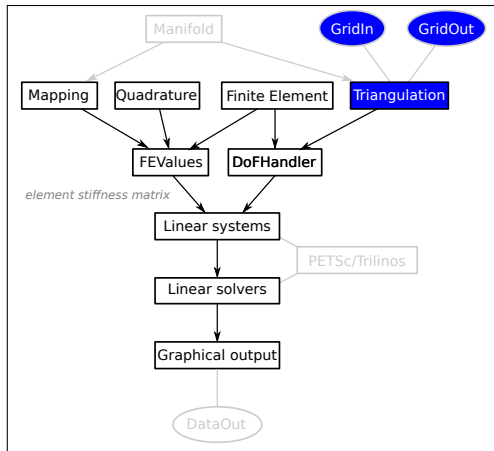
- ▶ mesh handling
- ▶ finite elements
- ▶ quadrature rules
- ▶ mapping rules
- ▶ assembly procedure
- ▶ linear solver

deal.II main modules →



Part 4:

Mesh handling in deal.II



- ▶ meshes are called `Triangulation`
- ▶ meshes can be created with functions in the `GridGenerator` namespace →
- ▶ meshes can be read from files
- ▶ meshes can be written to files

```
#include <deal.II/grid/tria.h>

int main()
{
    using namespace dealii;

    Triangulation<2> tria;
    GridGenerator::subdivided_hyper_cube(tria, 3);

    // output properties
    std::cout << tria.n_cells() << std::endl;
}
```

more information about `Triangulation` and `GridGenerator`:
<https://www.dealii.org/developer/doxygen/deal.II/index.html>

- ▶ meshes consist of cells
- ▶ it is possible to loop over all cells of a mesh
- ▶ cell properties (e.g., material ID) can be get/set →

see: CellAccessor

- ▶ vertices, lines, and faces of cells can be accessed

see: TriaAccessor

```
#include <deal.II/grid/tria.h>

int main()
{
    using namespace dealii;

    Triangulation<2> tria;
    // ...

    for(auto & cell : tria.active_cell_iterators())
    {
        std::cout << cell->material_id() << std::endl;
        std::cout << cell->face(0)->boundary_id()
                    << std::endl;
    }
}
```

note “operator-> ()”: we are working with iterators (here: TriaIterator)

- ▶ read the mesh “beam.msh” with `GridIn::read()`
- ▶ write the mesh to “task-1a-grid.vtk” with `GridOut::write_vtk()`
- ▶ take a look at the mesh in Paraview
- ▶ what properties are visualized?

... don't forget to include the needed header files!

Optional:

- ▶ write the mesh to “task-1a-data.vtk” with `DataOut::write_vtk()`
- ▶ create a mesh in Gmsh
- ▶ try out other mesh formats

Getting started with Linux:

- ▶ open a terminal and get/compile the code:

```
git clone https://github.com/peterrum/dealii-hereon-workshop.git
cd dealii-hzg-workshop-draft
cmake .
make task-1a-empty
```

- ▶ run the program:

```
./task-1a-empty
```

- ▶ for visualization use Paraview:

```
paraview
```

... in a new tab or new terminal

Loop over all cells and boundary faces and

- ▶ print the center of each cell
- ▶ assign material IDs to cells
- ▶ assign boundary IDs to faces

... hint: check results in Paraview!