FEniCS(x): Automated solution of PDEs by the FEM

What is FEniCS(x)?

FEniCS(x) (FE: Finite Elements, CS: Computational Software) is a software project that aims for the automated solution of partial differential equations (PDEs) by using the finite element method (FEM). The project targets mainly mathematicians that have already experience with PDEs and different FEM approaches, but do not want to develop an efficient FEM solver from scratch. An user can simply describe the weak formulation of the PDE and the desired Finite Element specifications and then FEniCS(x) handles the entire assembly and solution process in an optimized way.

FEniCS(x) also supports advanced techniques like saddle point problems, Brezzi-Douglas-Marini Finite Elements or enriched/extended Finite Elements, which are not available in commonly used FEM solvers. FEniCS(x) is actively used in research as Google Scholar lists about 3000 citations to the FEniCS book. Applications range from solid mechanics, fluid mechanics and FSI to electromagnetics - basically any subject where PDEs are involved. A great advantage of FEniCS(x) over commercial FEM solvers is that the user always has full control over the PDE models, the Finite Element discretization and the solver configuration.

Development of FEniCS started in 2003 as a collaboration between the Chalmers University of Technology and the University of Chicago. Nowadays, the development is mainly driven by the mathematicians Garth Wells and Andres Logg.

In 2019, the FEniCS(x) team started a refactoring process of the whole software. The new edition is called FEniCSx and published on GitHub. The last stable release of the legacy FEniCS is version 2019.1.0, while FEniCSx has currently the version 0.5.2 (as of November 2022) and is still in active development. Some features of legacy FEniCS are missing in FEniCSx or still experimental and the documentation is not complete yet. In particular, an overview of changes from FEniCS to FEniCSx is not available and an open issue on GitHub. However, the developers recommend to use FEniCSx:

> Now that development is focussed on FEniCSx, updates are made very rarely to the legacy FEniCS library. We recommend that users consider using FEniCSx instead of the legacy library.

How does it work?

FEniCS(x) has the goal to automate the whole workflow of the numerical solution of a PDE. The user provides the weak formulation and defines the solution spaces. Then, FEniCS automatically generates efficient C++ code to assemble the Finite Element system and solves it using a linear algebra backend like PETSc. The generated code supports parallelization using MPI.

FEniCSx is not a monolithic software system, but rather a combination of several building blocks that need to be used together. It consists of the following libraries:

- UFL: Unified Form Language that is used to express the weak formulation of a PDE in Python code
- FFCx: FEniCSx Form Compiler that compiles the weak formulation in UFL to efficient C++ code
- Basix: Provides the Finite Element basis functions for a wide range of Finite Elements (in legacy FEniCS this was provided by FIAT)
- DOLFINx: Contains all necessary data structures, connects the several parts of FEniCS and provides a unifying Python interface

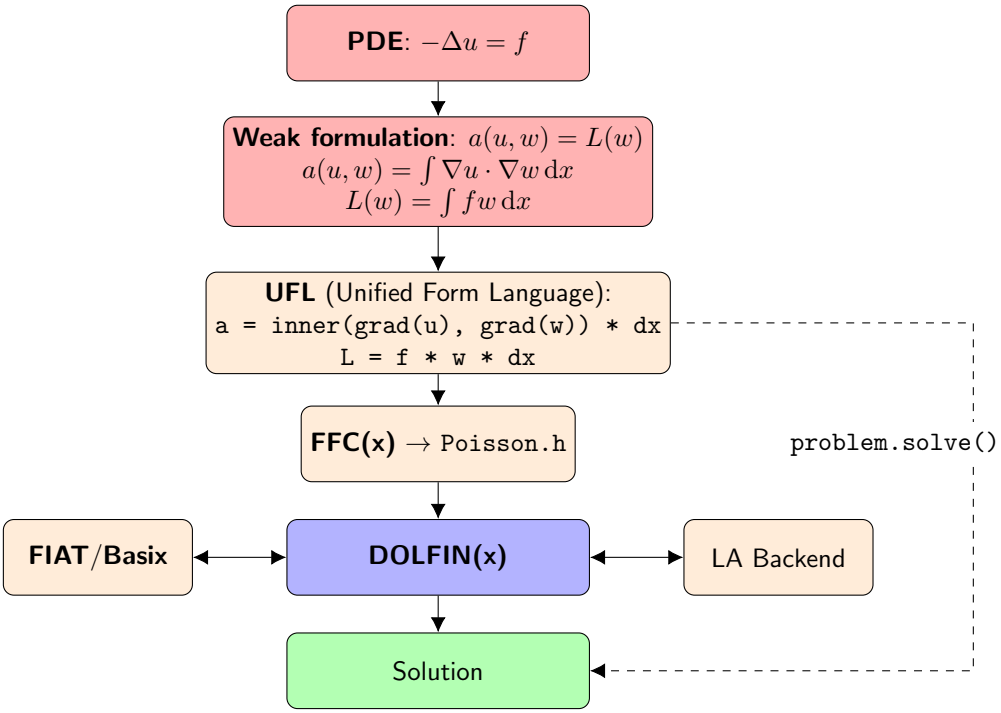

Figure 1: Overview of the FEniCS software project

FEniCSx employs mixed language programming by providing a convenient Python interface and performing the costly computations in C++. The user only needs to call the `solve` function of the Python interface and a JIT compiler handles everything under the hood.

Installation

There are multiple options to obtain a FEniCSx installation:

- Docker container: easiest option

```
docker run -ti dolfinx/dolfinx:stable
```

- Binary packages: easy, but usually not the latest version
- Built from source: This is a bit cumbersome since all parts of FEniCSx have to be built from source separately before compiling DOLFINx. I collected all necessary steps on Ubuntu in an installation script.

Getting started

There are really good tutorials for FEniCSx and legacy FEniCS. I would recommend starting with the Poisson problem and then solving real world problems (e.g. linear elasticity).

Tutorials

- "The FEniCSx Tutorial" by Jørgen S. Dokken (for FEniCSx)
- Solving PDEs in Python: The FEniCS Tutorial I by Hans P. Langtangen and Anders Logg (for legacy FEniCS)

Further information

- Documentation of the DOLFINx Python interface
- Documentation of the DOLFINx C++ API
- Automated Solution of Differential Equations by the Finite Element Method - "The FEniCS Book" by Anders Logg, Kent-Andre Mardal, Garth Wells (for legacy FEniCS)
- Roadmap of the FEniCSx project