Solving the Poisson equation

Authors: Hans Petter Langtangen, Anders Logg

Adapted to FEniCSx by Jørgen S. Dokken

The goal of this tutorial is to solve one of the most basic PDEs, the Poisson equations, with a few lines of code in FEniCSx. We start by introducing the most fundamental FEniCSx objects, such as `Function`, `FunctionSpace`, `TrialFunction` and `TestFunction`, and learn how to write a basic PDE solver. This will include:

- How to formulate a mathematical variational problem
- How to apply boundary conditions
- How to solve the discrete linear system
- How to visualize the solution

The Poisson equation is the following boundary-value problem

$$\begin{aligned} -\nabla^2 u(\mathbf{x}) &= f(\mathbf{x}) \qquad && \mathbf{x} \in \Omega \\ u(\mathbf{x}) &= u_D(\mathbf{x}) \qquad && \mathbf{x} \in \partial\Omega \end{aligned} \tag{1}$$

Here, $u = u(\mathbf{x})$ is the unknown function, $f = f(\mathbf{x})$ is a prescribed function, ∇^2 the Laplace operator, often written as Δ, Ω the spatial domain, and $\partial\Omega$ is the boundary of Ω. The Poisson problem, including both the PDE $-\nabla^2 u = f$ and the boundary condition $u = u_D$ on $\partial\Omega$, is an example of a *boundary-value problem*, which must be precisely state before it makes sense to start solving it numerically with FEniCSx.

In the two dimensional space with coordinates x and y, we can expand the Poisson equation as

$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f(x,y)$

The unknown u is now a function of two variables, $u = u(x,y)$, defined over the two-dimensional domain Ω.

The Poisson equation arises in numerous physical contexts, including heat conduction, electrostatics, diffusion of substances, twisting of elastic rods, inviscid fluid flow, and water waves. Moreover, the equation appears in numerical splitting strategies for more complicated systems of PDEs, in particular the Navier–Stokes equations.

Solving a boundary value problem in FEniCSx consists of the following steps:

1. Identify the computational domain Ω, the PDE, and its corresponding boundary conditions and source terms f.
2. Reformulate the PDE as a finite element variational problem.
3. Write a Python program defining the computational domain, the boundary conditions, the variational problem and the source terms, using FEniCSx.
4. Run the Python program to solve the boundary-value problem. Optionally, you can extend the program to derive quantities such as fluxes and averages, and visualize the results.

As we have already covered step 1, we shall now cover step 2-4.

Finite element variational formulation

FEniCSx is based on the finite element method, which is a general and efficient mathematical machinery for the numerical solution of PDEs. The starting point for the finite element methods is a PDE expressed in *variational form*. For readers not familiar with variational problems, it is suggested to reading a proper book on the finite element method in addition, as this tutorial is meant as a brief introduction to the subject. See the original tutorial [LL16] (Chapter 1.6.2).

The basic recipe for turning a PDE into a variational problem is:

- Multiply the PDE by a function v
- Integrate the resulting equation over the domain Ω
- Perform integration by parts of terms with second order derivatives.

The function v which multiplies the PDE is called a *test function*. The unknown function u that is to be approximated is referred to as a *trial function*. The terms trial and test functions are used in FEniCSx too. The test and trial functions belong to certain *function spaces* that specify the properties of the functions.

In the present case, we multiply the equation by a test function v and integrate over Ω:

$$\int_\Omega -\nabla^2 u v \mathrm{d}x = \int_\Omega f v \, \mathrm{d}x.$$

Here $\mathrm{d}x$ denotes the differential element for integration over the domain Ω. We will later let $\mathrm{d}s$ denote the differential element for integration over the boundary of Ω.

A rule of thumb is that when we derive variational formulations is that one tries to keep the order of derivatives of u and v as small as possible. Here, we have a second-order differential of u, which can be transformed to a first derivative by employing the technique of integration by parts. The formula reads

$$-\int_\Omega (\nabla^2 u) v \, \mathrm{d}x = \int_\Omega \nabla u \cdot \nabla v \mathrm{d}x - \int_{\partial\Omega} \frac{\partial u}{\partial n} v \, \mathrm{d}s,$$

where $\frac{\partial u}{\partial n} = \nabla u \cdot n$ is the derivative of u in the outward normal direction n on the boundary.

Another feature of variational formulations is that the test function v is required to vanish on the parts of the boundary where the solution u is known. See for instance [LM19].

In the present problem, this means that v is 0 on the whole boundary $\partial\Omega$. Thus, the second term in the integration by parts formula is zero, and we have that

$$\int_\Omega \nabla u \cdot \nabla v \mathrm{d}x = \int_\Omega f v \, \mathrm{d}x.$$

If we require that this equation holds for all test functions v in some suitable space $\hat{V}$, the so-called *test space*, we obtain a well-defined mathematical problem that uniquely determines the solution u which lies in some function space V. Note that V does not have to be the same space as $\hat{V}$. We call the space V the *trial space*. We refer to the equation above as the *weak form/variational form* of the original boundary value problem. We now properly state our variational problem: Find $u \in V$ such that

$$\int_\Omega \nabla u \cdot \nabla v \mathrm{d}x = \int_\Omega f v \mathrm{d}x \qquad \forall v \in \hat{V}.$$

For the present problem, the test and trial spaces V and $\hat{V}$ is defined as

$$\begin{aligned} V &= \{v \in H^1(\Omega) | v = u_D && \text{on } \partial\Omega\}, \\ \hat{V} &= \{v \in H^1(\Omega) | v = 0 && \text{on } \partial\Omega\}. \end{aligned} \tag{2}$$

In short, $H^1(\Omega)$ is the Sobolev space containing functions v such that v^2 and $|\nabla v|^2$ have finite integrals over Ω. The solution of the underlying PDE must lie in a function space where the derivatives are also continuous, but the Sobolev space $H^1(\Omega)$ allows functions with discontinuous derivatives. This weaker continuity requirement in our weak formulation (caused by the integration by parts) is of great importance when it comes to constructing the finite element function space. In particular, it allows the use of piecewise polynomial function spaces. This means that the function spaces are constructed by stitching together polynomial functions on simple domains such as intervals, triangles, quadrilaterals, tetrahedra and hexahedra.

The variational problem is a *continuous problem*: it defines the solution u in the infinite-dimensional function space V. The finite element method for the Poisson equation finds an approximate solution of the variational problem by replacing the infinite-dimensional function spaces V and $\hat{V}$ by *discrete* (finite dimensional) trial and test spaces $V_h \subset V$ and $\hat{V}_h \subset \hat{V}$. The discrete variational problem reads: Find $u_h \in V_h$ such that

$$\int_\Omega \nabla u_h \cdot \nabla v \,\mathrm{d}x = \int_\Omega fv \,\mathrm{d}x \qquad \forall v \in \hat{V}_h. \tag{3}$$

This variational problem, together with suitable definitions of V_h and $\hat{V}_h$ uniquely define our approximate numerical solution of the Poisson equation. Note that the boundary condition is encoded as part of the test and trial spaces. This might seem complicated at first glance, but means that the finite element variational problem and the continuous variational problem looks the same.

Abstract finite element variational formulation

We will introduce the following notations for variational problems: Find $u \in V$ such that

$$a(u, v) = L(v) \qquad \forall v \in \hat{V}. \tag{4}$$

For the Poisson equation, we have:

$$\begin{aligned} a(u, v) &= \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}x, \\ L(v) &= \int_\Omega fv \,\mathrm{d}x. \end{aligned} \tag{5}$$

From literature $a(u, v)$ is known as the *bilinear form* and $L(v)$ as a *linear form*. For every linear problem, we will identify all terms with the unknown u and collect them in $a(u, v)$, and collect all terms with only known functions in $L(v)$.

To solve a linear PDE in FEniCSx, such as the Poisson equation, a user thus needs to perform two steps:

1. Choose the finite element spaces V and $\hat{V}$ by specifying the domain (the mesh) and the type of function space (polynomial degree and type).

2. Express the PDE as a (discrete) variational problem: Find $u \in V$ such that $a(u, v) = L(v)$ for all $v \in \hat{V}$.

References

[LM19] Hans Petter Langtangen and Kent-Andre Mardal. *Introduction to Numerical Methods for Variational Problems*. Springer International Publishing, Cham, 2019. ISBN 978-3-030-23788-2. doi:10.1007/978-3-030-23788-2_1.