29th Nordic Seminar on Computational Mechanics
NSCM-29
R. Larsson (Ed.)
2016

OPTIMIZATION PROBLEMS ON DYNAMICAL DOMAINS WITH NON-MATCHING MESHES

JØRGEN S. DOKKEN, SIMON W. FUNKE, AUGUST JOHANSSON

Simula Research Laboratory
Martin Linges vei 25, 1364 Fornebu, Norway
e-mail: dokken@simula.no, web page: www.linkedin.com/in/jorgen-dokken

Key words: Optimization, Nitsche, Finite Element Method,
Adjoint, Dynamic Domains, Non-matching meshes.

Summary. An approach for solving physics-based optimization problems on dynamical domains is presented. The optimization problems of interest are those where the goal quantity is a functional and the constraints are sets of partial differential equations. Each domain may have its own mesh and a Nitsche based finite element method will be used to discretize the equations and enforce continuity over the interfaces. This method will be applied to an optimal Poisson problem, which is solved using the software FEniCS, dolfin-adjoint and moola.

1 MOTIVATION

There are many problems in industry which can be represented by optimization problems. One example is how to reduce the drag over an airfoil while maximizing the lift. In this example the lift and drag are goal quantities which we want to minimize or maximize. Finding the drag and lift for one specific wing design can be done by representing the physical system as a set of partial differential equations (PDEs). Solving such equations numerically can be done with the Finite Element Method (FEM).

Another example is the optimization of wind or tidal turbines. Tidal-stream turbines are one of the most promising technologies to extract tidal stream energy. It is therefore important to find the optimal configuration of such turbines. Since these turbines consists of a set of rotating blades, the domain modeled should be representeded by a dynamical domain.

Currently methods such as the Arbitrary Lagrangian-Eulerian (ALE) formulation[1] have been used to model such domains. With ALE, the mesh follows the dynamics of the domain. If we want to rotate the turbines without constrictions, the mesh would experience large deformations and degenerate quality. Therefore fixing such deformations can be done with smoothing or full re-meshing. However, in an optimization setting, re-meshing is a non-differentiable operation and can be problematic in gradient based optimization. Gradient based optimization is preferred since it often reduces the number of functional evaluations significantly compared to gradient free methods. We therefore seek another way of handling dynamical domains. The approach chosen is to allow each domain to have its own mesh and handle the interface conditions over the non-matching meshes using Nitsche's method[2]. Then, the domains can change without the need of remeshing.

2 OPTIMIZATION AND THE ADJOINT MODEL

We consider an optimization problem with one goal functional and a set of PDE constraints

$$\min_{u,m} \quad J(u(m), m), \tag{1}$$

$$\text{subject to } F_i(u(m), m) = 0, \quad i = 1, \ldots, N, \tag{2}$$

where $m = (m_1, m_2, \ldots, m_M)$ is the control quantities, u is the solution of the set of PDEs $F_i(u, m) = 0, \quad i = 1, \ldots, N$ and J is the goal quantity.

Since the goal quantity may be an indirect function of the control variables m, we create a new function $\hat{J}(m) = J(u(m), m)$. This converts the constrained problem to an unconstrained problem. The new optimization problem is

$$\min_{m} \hat{J}(m). \tag{3}$$

When finding the minimum of this functional we need to evaluate it at different points in the space of the design parameter. Since this evaluation involves solving a PDE that may be computationally expensive, we want to reduce the number of such evaluations. To do this we apply gradient based optimization algorithms and therefore we need information about the gradient of $\hat{J}$ with respect to the design parameters m. By applying the chain rule to the gradient

$$\frac{\mathrm{d}\hat{J}}{\mathrm{d}m} = \frac{\partial J}{\partial u}\frac{\mathrm{d}u}{\mathrm{d}m} + \frac{\partial J}{\partial m}. \tag{4}$$

In this equation the Jacobian $\mathrm{d}u/\mathrm{d}m$ is rather hard to compute, since it is a dense matrix of dimensions (solution space $\times$ parameter space). We use the adjoint approach for avoiding computation of $\mathrm{d}u/\mathrm{d}m$. This approach starts by considering the set of PDE-constraints $F(u, m) = 0$ and take the total derivative of both sides of this equation

$$\frac{\mathrm{d}F}{\mathrm{d}m} = \frac{\partial F}{\partial u}\frac{\mathrm{d}u}{\mathrm{d}m} + \frac{\partial F}{\partial m} = \frac{\mathrm{d}0}{\mathrm{d}m} = 0. \tag{5}$$

Suppose that $\partial F/\partial u$ is invertible and insert Equation (5) into Equation (4), obtaining

$$\frac{\mathrm{d}\hat{J}}{\mathrm{d}m} = -\frac{\partial J}{\partial u}\left(\frac{\partial F}{\partial u}\right)^{-1}\frac{\partial F}{\partial m} + \frac{\partial J}{\partial m}. \tag{6}$$

By taking the Hermitian transpose of Equation (6) and defining the the adjoint variable λ to be the solution of the Jacobian acting on a vector we get the following system

$$\frac{\mathrm{d}\hat{J}}{\mathrm{d}m}^* = -\frac{\partial F}{\partial m}^* \lambda + \frac{\partial J}{\partial m}^*, \tag{7}$$

$$\left(\frac{\partial F}{\partial u}\right)^* \lambda = \frac{\partial J}{\partial u}^*. \tag{8}$$

For finding the solution to Equation (7) we first solve the adjoint equation (8) for λ, then taking its inner product with $(\partial F/\partial m)^*$ and finally adding $(\partial J/\partial m)^*$. When solving the adjoint equation, we need to specify the functional of interest, but we do not need to specify the design parameter m. Therefore solving the adjoint equation is very efficient when we have a small number of functionals and a large number of design parameters.

3 DYNAMIC DOMAINS AND NITSCHE FINITE ELEMENT METHOD

Consider Figure 1 illustrating a turbine with two non-matching meshes: the background mesh $\mathcal{T}_1$ and turbine-fitted mesh $\mathcal{T}_2$. Note that parts of $\mathcal{T}_2$ overlap $\mathcal{T}_1$. To illustrate the concept of dynamic domains we will rotate $\mathcal{T}_2$ with a prescribed velocity ω. Therefore, both $\mathcal{T}_1$ and $\mathcal{T}_2$ will be time-dependent, $\mathcal{T}_1 = \mathcal{T}_1(t)$ and $\mathcal{T}_2 = \mathcal{T}_2(t)$. Note that $\mathcal{T}_2$ will have the same topology since only the location of the vertices will change. For $\mathcal{T}_1$, different vertices may become involved in the discretization as $\mathcal{T}_2$ move. On $\mathcal{T}_i$ we will construct finite element spaces, $V_{h,i}(t)$, $i = 1, 2$ in a quasi-stationary manner.

Of particular interest is the interface $\Gamma(t) = \partial\mathcal{T}_2(t) \cap \mathcal{T}_1(t)$ on which we need to enforce continuity. This is done weakly using a Nitsche formulation. For example, if we consider a single Poisson problem $F(u(m), m) = -\Delta u - f(m)$ in the domain $\Omega = \mathcal{T}_1(t) \cup \mathcal{T}_2(t)$, we seek $[u] = [\mathbf{n} \cdot \nabla u] = 0$ across $\Gamma(t)$. This continuity condition can be enforced weakly using a Nitsche FEM as described in[2]: For each t, find $u_h \in V_h(t) = V_{h,1}(t) \times V_{h,2}(t)$ such that $a_h(u_h, v_h) = L(v_h)$ for all $v_h \in V_h(t)$, and

$$a_h(u_h, v_h) = \sum_{i=1}^{2} (\nabla u_h, \nabla v_h)_{\mathcal{T}_i} - (\mathbf{n} \cdot \nabla u_h, [v_h])_\Gamma - (\mathbf{n} \cdot \nabla v_h, [u_h])_\Gamma + \gamma(h^{-1}[u_h], [v_h])_\Gamma \tag{9}$$

$$L(v_h) = \sum_{i=1}^{2} (f, v_i)_{\mathcal{T}_i}, \tag{10}$$

where γ is a positive constant and $u_h = (u_{h,1}, u_{h,2})$, $v_h = (v_{h,1}, v_{h,2})$.

4 IMPLEMENTATION

FEniCS[3] has been used for solving the PDEs related to optimization constraints as well as solving the heat equation. The technology for handling interfaces between non-matcing meshes in FEniCS is called MultiMesh. For generating the adjoint equations for the optimization problem we have extended dolfin-adjoint[4] to support non-matching meshes. Dolfin-adjoint is a framework for deriving the automated adjoint equations. For solving the optimization problem we have used moola, an optimization framework for dolfin-adjoint and FEniCS. The combination of these frameworks will allow the solution of complex PDE-constrained optimization problems with compact of high-level Python code. Results for optimization of the source term of the Poisson equation for a stationary domain will be presented, as well as solution of the heat equation with dynamical domains.

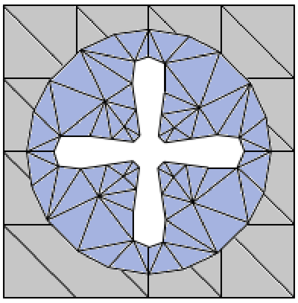

Figure 1: An illustration of a turbine represented with non-fitted meshes. The turbine can be rotated without remeshing. The blue part is $\mathcal{T}_2$, the grey part is $\mathcal{T}_1$.

REFERENCES

[1] Donea, J., Huerta, A., Ponthot, J.-P. & Rodriguez-Ferran, A. Encyclopedia of computational mechanics vol. 1: Fundamentals., chapter 14: Arbitrary Lagrangian-Eulerian Methods (2004).

[2] Hansbo, P. Nitsche's method for interface problems in computational mechanics. *GAMM-Mitteilungen* **28**, 183–206 (2005).

[3] Logg, A., Mardal, K.-A. & Wells, G. *Automated solution of differential equations by the finite element method: The FEniCS book*, vol. 84 (Springer Science & Business Media, 2012).

[4] Farrell, P. E., Ham, D. A., Funke, S. W. & Rognes, M. E. Automated derivation of the adjoint of high-level transient finite element programs. *SIAM Journal on Scientific Computing* **35**, C369–C393 (2013).