PDE-constrained optimisation

Problem statement

Let m be a vector of some parameters. For example, m might be the values of an initial condition, or of a source term, or of a boundary condition.

Let $F(u, m) \equiv 0$ be a (system of) partial differential equations that describe the physics of the problem of interest. F is a vector expression (one entry for each equation), with all terms in the equation gathered on to the left-hand side. The idea is that, for any (feasible) choice of $m \in \mathbb{R}^M$, the PDE F can be solved to yield the solution $u \in \mathbb{R}^U$. In other words, *the solution u can be thought of as an implicit function $u(m)$ of the parameters m*, related through the PDE $F(u, m) \equiv 0$. We never have an explicit expression for u in terms of m, but as we shall see, we can still discuss its derivative $\mathrm{d}u/\mathrm{d}m$.

Feasibility

In optimisation, to say that a point m in parameter space is *feasible* means that it satisfies all of the constraints on the choice of parameters. To say that the pair (u, m) is feasible means that m is feasible, *and* that u satisfies the relationship $F(u, m) = 0$.

If the problem $F(u, m)$ is time-dependent, this abstraction still holds. In this case, think of u as a vector containing all time values of all prognostic variables. In the discrete case, u is a vector with the value of the solution at the first timestep, then the value at the second timestep, and so on, for however many timesteps are required.

Finally, let $J(u, m)$ be a *functional* of interest. J represents the quantity to be optimised: for example, the quality of a design is to be maximised, or the misfit between observations and computations is to be minimised.

A general statement of the PDE-constrained optimisation problem is then given as follows: find the m that minimises $J(u, m)$, subject to the constraint that $F(u, m) = 0$. For simplicity, we suppose that there are no further constraints on the choice of m; there are well-known techniques for handling such situations. If J is to be maximised instead of minimised, just consider minimising the functional $-J$.

Throughout this introduction, we shall implicitly consider the case where *the dimension of the parameter space is very large*. This means that we shall seek out algorithms that scale well with the dimension of the parameter space, and discard those that do not. We shall also generally assume that *solving the PDE is very expensive*: therefore, we will seek out algorithms which attempt to minimise the number of PDE solutions required. This combination of events – a large parameter space, and an expensive PDE – is the most interesting, common, practical and difficult situation, and therefore it is the one we shall attempt to tackle head-on.

Functional

A *functional* is a function that acts on some vector space, and *returns a single scalar number*.

Solution approaches

There are many ways to approach solving this problem. The approach that we shall take here is to apply a *gradient-based optimisation algorithm*, as these techniques scale to large numbers of parameters and to complex, nonlinear, time-dependent PDE constraints.

To apply an optimisation algorithm, we will convert the PDE-constrained optimisation problem into an unconstrained optimisation problem. Let $\widehat{J}(m) \equiv J(u(m), m)$ be the functional *considered as a pure function of the parameters m*: that is, to compute $\widehat{J}(m)$, solve the PDE $F(u, m) = 0$ for u, and then evaluate $J(u, m)$. The functional $\widehat{J}$ has the PDE constraint "built in": by considering $\widehat{J}$ instead of J, we convert the constrained optimisation problem to a simpler, unconstrained one. The problem is now posed as: find the m that minimises $\widehat{J}(m)$.

Given some software that solves the PDE $F(u, m) = 0$, we have a black box for computing the value of the functional $\widehat{J}$, given some argument m. If we can only evaluate the functional, and have no information about its derivatives, then we are forced to use a gradient-free optimisation algorithm such as a genetic algorithm. The drawback of such methods is that they typically scale very poorly with the dimension of the parameter space: even for a moderate sized parameter space, a gradient-free algorithm will typically take hundreds or thousands of functional evaluations before terminating. Since each functional evaluation involves a costly PDE solve, such an approach quickly becomes impractical

By contrast, optimisation algorithms that can exploit information about the derivatives of $\widehat{J}$ can typically converge onto a

Other approaches

local minimum with one or two orders of magnitude fewer iterations, as the gradient provides information about where to step next in parameter space. Therefore, if evaluating the PDE solution is expensive (and it usually is), then computing derivative information of $\widehat{J}$ becomes very important for the practical solution of such PDE-constrained optimisation problems.

So, how should the gradient $\mathrm{d}\widehat{J}/\mathrm{d}m$ be computed? There are three main approaches, each with their own advantages and disadvantages. Discussing these strategies is the topic of the next section.

References

[2M-Kar39] W. Karush. Minima of functions of several variables with inequalities as side constraints. Master's thesis, University of Chicago, Chicago, IL, USA, 1939.

[2M-KT51] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of 2nd Berkeley Symposium*, 481–492. University of California Press, 1951.

No discussion of PDE-constrained optimisation would be complete without mentioning the "oneshot" approach. Instead of starting with some initial guess m and applying an optimisation algorithm, the oneshot approach derives auxiliary equations that provide necessary and sufficient conditions for finding an optimum. These coupled equations are then solved, almost always with a matrix-free approach. The necessary and sufficient conditions are referred to as the KKT conditions, and the system referred to as the KKT system, after Karush, Kuhn and Tucker, the mathematicians who derived the optimality system [2M-Kar39] [2M-KT51]. Interestingly, one of the equations in the KKT system is the adjoint equation, which will be derived in a different way in the next section.