Interior Control of the Heat Equation with OpenFOAM

Víctor Hernández-Santamaría * José Vicente Lorenzo Gómez †

June 26, 2018

Let us consider the distributed control of the heat equation through the minimization problem

$$\begin{aligned}\min_{u\in L^2(\Omega\times(0,T))} \mathcal{J}\left(u, y\left(u\right)\right) = \min_{u\in L^2(\Omega\times(0,T))} \frac{\beta_1}{2}\int_0^T\int_\Omega u^2 \ \mathrm{d}\Omega \ \mathrm{d}t + \frac{\beta_2}{2}\int_0^T\int_\Omega \left(y-y_d\right)^2 \ \mathrm{d}\Omega \ \mathrm{d}t \\ + \frac{\beta_3}{2}\int_\Omega \left(y\left(\cdot,T\right)-Y_d\right)^2 \ \mathrm{d}\Omega, \quad \beta_1,\ \beta_2,\ \beta_3 \in \mathbb{R}^+,\end{aligned} \tag{1}$$

where u is the control, y is the state variable, y_d is the target state in $\Omega\times(0,T)$, and Y_d is the target state at time T. The optimization problem is subject to the state equation,

$$\begin{cases} \partial_t y - \Delta y = f + u & \text{in } Q = \Omega\times(0,T), \\ y = y_D & \text{on } \Sigma_D = \Gamma_D\times(0,T), \\ \dfrac{\partial y}{\partial n} = y_N & \text{on } \Sigma_N = \Gamma_N\times(0,T), \\ y\left(\cdot,0\right) = y_0 & \text{in } \Omega, \end{cases} \tag{2}$$

with $\Gamma = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$.

Since the problem is linear the solution can be separated into a part that depends on the known source term, boundary conditions, and initial condition; and a control-dependent term as

$$y = y_f + y_u, \tag{3}$$

where

$$\begin{cases} \partial_t y_f - \Delta y_f = f & \text{in } Q, \\ y_f = y_D & \text{on } \Sigma_D, \\ \dfrac{\partial y_f}{\partial n} = y_N & \text{on } \Sigma_N, \\ y_f\left(\cdot,0\right) = y_0 & \text{in } \Omega. \end{cases} \tag{4}$$

and

$$\begin{cases} \partial_t y_u - \Delta y_u = u & \text{in } Q, \\ y_u = 0 & \text{on } \Sigma_D, \\ \dfrac{\partial y_u}{\partial n} = 0 & \text{on } \Sigma_N, \\ y_u\left(\cdot,0\right) = 0 & \text{in } \Omega. \end{cases} \tag{5}$$

Let us multiply the PDE in (5) by φ in $L^2\left(Q\right)$,

$$\int\int_Q \varphi\left(\partial_t y_u - \Delta y_u - u\right) \ \mathrm{d}\Omega \ \mathrm{d}t = 0, \quad \forall\varphi \in L^2\left(Q\right),$$

*DeustoTech, University of Deusto.
†Universidad Autónoma de Madrid.

and integrate by parts twice,

$$\int_\Omega \left(\varphi(T)\, y_u(T) - \varphi(0) \underbrace{y_u(0)}_{=0} \right) \mathrm{d}\Omega - \int\int_Q y_u \left(\partial_t \varphi + \Delta y_u \right) \mathrm{d}\Omega\, \mathrm{d}t - \int\int_Q \varphi\, u\, \mathrm{d}\Omega\, \mathrm{d}t$$
$$- \int\int_\Sigma \varphi \underbrace{\frac{\partial y_u}{\partial n}}_{=0 \text{ on } \Sigma_N} \mathrm{d}\Gamma\, \mathrm{d}t + \int\int_\Sigma \underbrace{y_u}_{=0 \text{ on } \Sigma_D} \frac{\partial \varphi}{\partial n} \mathrm{d}\Gamma\, \mathrm{d}t = 0.$$

One can get rid of the boundary terms by setting

$$\begin{cases} \varphi = 0, & \text{on } \Sigma_D, \\ \dfrac{\partial \varphi}{\partial n} = 0, & \text{on } \Sigma_N. \end{cases} \tag{6}$$

Thus,

$$\left(\varphi(\cdot,T), y_u(\cdot,T)\right)_{L^2(\Omega)} + \left(-\partial_t \varphi - \Delta\varphi, y_u\right)_{L^2(Q)} = (\varphi, u)_{L^2(Q)}. \tag{7}$$

Moreover, by developing the terms in the cost function one gets

$$\begin{aligned} \mathcal{J} =& \frac{1}{2}\left[\beta_3 \left(y_u(\cdot,T), y_u(\cdot,T)\right)_{L^2(\Omega)} + \beta_2 \left(y_u, y_u\right)_{L^2(Q)}\right] \\ & - \left[\beta_3 \left(Y_d - y_f(\cdot,T), y_u(\cdot,T)\right)_{L^2(\Omega)} + \beta_2 \left(y_d - y_f, y_u\right)_{L^2(Q)}\right] \\ & + \frac{1}{2}\left[\beta_3 \left(Y_d - y_f(\cdot,T), Y_d - y_f(\cdot,T)\right)_{L^2(\Omega)} + \beta_2 \left(y_d - y_f, y_d - y_f\right)_{L^2(Q)}\right] \\ & + \frac{\beta_1}{2}(u,u)_{L^2(Q)}. \end{aligned}$$

Let us introduce the space $H = L^2(\Omega) \times L^2(Q)$ and take $(\Phi_1, \phi_1) \in H$, $(\Phi_2, \phi_2) \in H$, we define the inner product in H as

$$\left[(\Phi_1, \phi_1), (\Phi_2, \phi_2)\right]_H = \beta_3 (\Phi_1, \Phi_2)_{L^2(\Omega)} + \beta_2 (\phi_1, \phi_2)_{L^2(Q)}, \tag{8}$$

so that (7) becomes

$$\left[\left(\frac{1}{\beta_3}\varphi(\cdot,T), \frac{1}{\beta_2}\left(-\partial_t\varphi - \Delta\varphi\right)\right), \left(y_u(\cdot,T), y_u\right)\right]_H = (\varphi, u)_{L^2(Q)}, \tag{9}$$

and the cost functional can be expressed as

$$\begin{aligned} \mathcal{J} =& \frac{1}{2}\left[\left(y_u(\cdot,T), y_u\right), \left(y_u(\cdot,T), y_u\right)\right]_H - \left[\left(Y_d - y_f(\cdot,T), y_d - y_f\right), \left(y_u(\cdot,T), y_u\right)\right]_H \\ & + \frac{1}{2}\left[\left(Y_d - y_f(\cdot,T), y_d - y_f\right), \left(Y_d - y_f(\cdot,T), y_d - y_f\right)\right]_H + \frac{\beta_1}{2}(u,u)_{L^2(Q)}. \end{aligned}$$

We now define a linear operator $\Lambda : L^2(Q) \to H$ that takes a control $u \in L^2(Q)$ and returns the state $\Lambda u = (y_u((\cdot,T), y_u) \in H$, this yields

$$\begin{aligned} \mathcal{J} =& \frac{1}{2}\left[\Lambda u, \Lambda u\right]_H + \frac{\beta_1}{2}(u,u)_{L^2(Q)} - \left[\left(Y_d - y_f(\cdot,T), y_d - y_f\right), \Lambda u\right]_H \\ & + \frac{1}{2}\left[\left(Y_d - y_f(\cdot,T), y_d - y_f\right), \left(Y_d - y_f(\cdot,T), y_d - y_f\right)\right]_H. \end{aligned} \tag{10}$$

By the definition of adjoint operator,

$$\left[(\Phi, \phi), \Lambda u\right]_H = \left(\Lambda^*(\Phi, \phi), u\right)_{L^2(\Omega\times(0,T))}, \quad \forall u \in L^2(Q), \quad \forall (\Phi, \phi) \in H,$$

and from (10) one concludes that $\Lambda^*\left(y_u\left(\cdot,T\right),y_u\right)=\varphi$ with φ solution to the adjoint problem

$$\begin{cases}-\partial_t\varphi-\Delta\varphi=\beta_2 y_u, & \text{in } Q,\\ \varphi\left(\cdot,T\right)=\beta_3 y_u\left(\cdot,T\right), & \text{in } \Omega,\\ \varphi=0, & \text{on } \Sigma_D,\\ \dfrac{\partial\varphi}{\partial n}=0, & \text{on } \Sigma_N,\end{cases} \tag{11}$$

where the boundary conditions come from (6).

Now the Gâteaux derivative of $\mathcal{J}$ with respect to the control u in the direction δu yields

$$\begin{aligned}\mathcal{D}_{\delta u}\mathcal{J}&=\left[\Lambda u,\Lambda\delta u\right]_H+\beta_1\left(u,\delta u\right)_{L^2(Q)}-\left[\left(Y_d-y_f\left(\cdot,T\right),y_d-y_f\right),\Lambda\delta u\right]_H=\\&=\left(\left(\Lambda^*\Lambda+\beta_1 I\right)u-\Lambda^*\left(Y_d-y_f\left(\cdot,T\right),y_d-y_f\right),\delta u\right)_{L^2(Q)},\end{aligned}$$

from where one can deduce the functional gradient that provides local information about how the control must be updated in order to make the cost decrease,

$$\mathcal{J}'\left(u\right)=\left(\Lambda^*\Lambda+\beta_1 I\right)u-\Lambda^*\left(Y_d-y_f\left(\cdot,T\right),y_d-y_f\right). \tag{12}$$

In the steepest descent method the cost gradient is used to update the control as

$$u^{(n+1)}=u^{(n)}-\epsilon\mathcal{J}'\left(u^{(n)}\right),\quad \epsilon\in\mathbb{R}^+,$$

with ϵ sufficiently small. This method, however, shows a rather slow convergence rate, and the value of ϵ is chosen arbitrarily. For these reasons it is advisable to use the conjugate gradient method instead, as it converges faster and not only gives the descent direction but the step size as well. In order to apply it we define

$$\begin{aligned}A_{cg}&=\Lambda^*\Lambda+\beta_1 I:L^2\left(Q\right)\to L^2\left(Q\right),\\ b_{cg}&=\Lambda^*\left(Y_d-y_f\left(\cdot,T\right),y_d-y_f\right)\in L^2\left(Q\right),\end{aligned}$$

so that the cost gradient reads as

$$\mathcal{J}'\left(u\right)=A_{cg}u-b_{cg}.$$

The steepest descent method has been coded with the C++ library OpenFOAM.