



HAL
open science

Investigating gas furnace control practices with reinforcement learning

M Renault, J Viquerat, P Meliga, G.-A Grandin, N Meynet, E Hachem

► **To cite this version:**

M Renault, J Viquerat, P Meliga, G.-A Grandin, N Meynet, et al.. Investigating gas furnace control practices with reinforcement learning. *International Journal of Heat and Mass Transfer*, 2023, 209, pp.124147. 10.1016/j.ijheatmasstransfer.2023.124147 . hal-04245154

HAL Id: hal-04245154

<https://hal.science/hal-04245154>

Submitted on 16 Oct 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Investigating gas furnace control practices with reinforcement learning

M. Renault^a, J. Viquerat^a, P. Meliga^a, G.-A. Grandin^b, N. Meynet^b, E. Hachem^{a,*}

^aMines Paris, PSL University, Centre for material forming (CEMEF), UMR CNRS, 06904 Sophia Antipolis, France

^bENGIE Lab CRIGEN, 4 rue Joséphine Baker 93240 Stains, France

Abstract

Gas furnaces are the most widely used means of heating in industry, and with the growing concern for environmental issues, and a global energy crisis at our doorstep, the optimization of the processes related to them becomes a key challenge. This paper aims at introducing a new way of practicing gas furnace control involving simulations, virtual sensors and deep reinforcement learning (DRL) techniques. In order to do so we designed a set of simulations of conjugate heat transfer systems governed by the coupled Navier–Stokes and heat equations for single-step control. The DRL algorithm used in this paper is the policy-based optimization (PBO) algorithm specialized in single-step (or open-loop) control. We explore its ability to find global maxima in different situations and under various constraints. Therefore, various 2D and 3D cases are tackled, in which the position of the work piece, the flow rate, and other parameters are controlled. The obtained results highlight the potential of the DRL framework combined with computational fluid dynamics (CFD) conjugate heat transfer systems for optimizing searches in large parameter spaces. For the 2D case, PPO achieved an increase of 89% in temperature homogeneity, and for the 3D case an increase of 7% in final temperature with the same total input.

Keywords: Deep Reinforcement Learning; Artificial Neural Networks; Conjugate heat transfer; Computational fluid dynamics; Thermal control; Serpentine.

1. Introduction

Just like cooling, properly heating a part is a challenge that manufacturers take up every day to obtain the desired properties in the part, whether they are mechanical, electrical, optical or aesthetic. To achieve that, gas furnaces have been studied for a long time and their control has evolved to arrive today at the cohabitation of many techniques, including manual control, fuzzy control [1], proportional-integral (PI)/proportional-integral-derivative (PID) single-loop control or cascade control [2, 3] among others. The reason for this flourishing of techniques is the difficulty to accurately perform temperature control, whether it is because of the difficulty to measure the temperature accurately inside the chamber, or because of the different characteristics of temperature variation through flow control such as non linearity, inertia and time delay.

Temperature control has been addressed by a lot of articles in the literature, and with the emergence of Artificial Neural Networks (ANNs) taking advantage of the most recent advances in computational power and data analysis, new control techniques that are more robust and can deal with non linearity, inertia and time delay have been developed : image processing and clustering to control gas burners [4, 5], recurrent neural networks (RNNs) with electrical furnaces [6, 7], neural networks for metal quality control [8] and for flame stability control [9–11], and radial basis function neural networks (RBFNNs) to control coke furnaces [12]. In the aerospace industry, burners are also very thoroughly studied to prevent any incident inside aircraft engines. They use neural networks in the active control of the burning chambers and find ways to prevent oscillating patterns to occur during combustion [13]. Such patterns are generally responsible for a premature

*elie.hachem@minesparis.psl.eu

21 fatigue of the materials composing the chamber, as well as disturbances to the good functioning
 22 of the combustion. This is also true for industrial gas furnaces ; even though these issues are less
 23 explored in the field of furnace control, they become more and more important with the use of
 24 mixed fuels, and the alternating use of natural gas and dihydrogen.

25 In the realm of optimal control with constraints on the state of the system, the combination of
 26 deep neural networks (DNNs) and reinforcement learning (RL) algorithms (a formal framework in
 27 which an agent learns by interacting with an environment and learns by gathering experience) has
 28 brought new cards to the table in such a way that the best performing algorithms in a wide variety
 29 of tasks (*e.g.*, games [14, 15], cooling control [16], autonomous cars [17], medicine [18], energy
 30 [19, 20]). In fluid dynamics, this so-called deep reinforcement learning (deep RL, or DRL) has also
 31 been used with success for flow control and shape optimization with success (Refs. [21, 22] and the
 32 references therein) by taking advantage of its robustness to non linearity and to high dimensional
 33 spaces. This field of application is still at an early stage, as evidenced by the scarce literature
 34 dedicated to DRL-based thermal control [23, 24], but it shows great promise for the future of fluid
 35 related topics.

36 This work aims at introducing DRL into the field of gas furnace control. More specifically, it
 37 assesses the performances of proximal policy optimization (PPO [25]) for the one-step control of a
 38 heating chamber. We use an algorithm introduced in [26] and whose relevance for open-loop flow
 39 control problems is assessed in [27] that is a degenerate version of the classical PPO algorithm. The
 40 choice for PPO is driven by its data efficiency (a decisive criteria for computationally expensive
 41 simulations), ease of implementation and already widely assessed performance. Several problems
 42 of conjugate heat transfer in two and three dimensions are used as testbed to push forward the
 43 development of this novel approach. To the best of the authors knowledge, this constitutes the
 44 first attempt to achieve DRL-based control of conjugate forced convection heating processes.

45 2. Governing equations for fluid mechanics

46 The focus of this research is on conjugate heat transfer and laminar, incompressible fluid flow
 47 problems in two and three-dimensions, for which the conservation of mass, momentum and energy
 48 is described by the nonlinear, coupled Navier–Stokes and heat equations

$$\nabla \cdot \mathbf{u} = 0, \tag{1}$$

$$\rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}) = \nabla \cdot (-p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u})), \tag{2}$$

$$\rho c_p(\partial_t T + \mathbf{u} \cdot \nabla T) = \nabla \cdot (\lambda \nabla T), \tag{3}$$

49 where \mathbf{u} is the velocity field, p is the pressure, T is the temperature, $\boldsymbol{\varepsilon}(\mathbf{u}) = (\nabla \mathbf{u} + \nabla \mathbf{u}^T)/2$ is the
 50 rate of deformation tensor, we assume here constant fluid density ρ , dynamic viscosity μ , thermal
 51 conductivity λ , and specific heat c_p , and we have neglected buoyancy and radiative heat transfer,
 52 on behalf of the focus being on conjugate forced convection heat transfer.

53 This is solved here with an in-house stabilized finite elements environment cast in the Variational
 54 Multiscale (VMS) framework. This allows using equal order linear approximations for all variables
 55 (very desirable due to its simplicity of implementation and affordable computing cost) by enhancing
 56 the stability of the Galerkin method via a series of additional derived residual based terms evaluated
 57 over element interior. The solid is treated as an immersed body, using the Immerse Volume Method
 58 to compute the amount of heat exchanged between the solid and the fluid only from the individual
 59 material properties on either side of it (which in turn removes the need for a heat transfer coefficient,
 60 a limiting issue for the present numerical experiments where we vary the position of the solid). For
 61 details about the numerical framework, including the interface capturing method used to generated
 62 strongly anisotropic meshes adapted at the fluid-solid boundary (to ensure that the fluid properties
 63 are distributed as accurately and smoothly as possible over the smallest possible thickness around
 64 the interface), the reader can refer to Refs. [24, 28]. The relevance of this numerical method has
 65 been validated on multiple benchmarks here [29] and in particular for heat transfer here [30]. For
 66 the sake of simplicity we will not discuss it here.

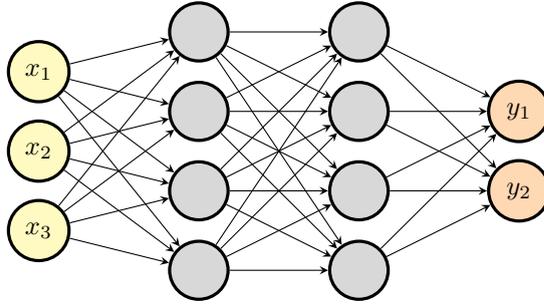


Figure 1: Fully connected neural network with two hidden layers, modeling a mapping from data living in \mathbb{R}^3 to data living in \mathbb{R}^2 .

3. Deep reinforcement learning and proximal policy optimization

3.1. Neural networks

A neural network (NN) is a connected collection of non-linear functions. It is extremely useful when trying to mimic the relationship between multiple highly non-linear phenomena. A fully connected network is sketched in figure 1, it has layers (represented as columns in the sketch) and within each of them the neurons are connected to all of those contained in the next layer. The input layer is the one that receives information from the outside, the output layer is the one that gives the result and between them are hidden layers. Building an efficient neural network requires a relevant architecture (e.g., type of network, depth, width of each layer), finely tuned hyper-parameters (i.e., parameters that cannot be learned directly by the neural network, e.g., optimizer, learning rate, batch size) and an adequate amount of data to learn from. More details here Ref. [31] and the references therein.

3.2. Deep reinforcement learning

Reinforcement learning (RL) is a type of machine learning in which an agent learns the actions to do in an environment in order to get the best reward possible. Such method is often formulated as a Markov Decision Process for which a full loop looks like this :

- Assume the environment is in state $s_t \in S$ at iteration t , where S is a set of states;
- The agent uses w_t , an observation of the current environment state (and possibly a partial subset of s_t) to take action $a_t \in A$, where A is a set of actions;
- The environment reacts to the action by transitioning from s_t to state $s_{t+1} \in S$;
- The agent is fed with a reward $r_t \in R$, where R is a set of rewards, and a new observation w_{t+1} .

This repeats until a steady state is reached. The succession of states and actions defines a trajectory $\tau = (s_0, a_0, s_1, a_1, \dots)$ for which the agent will try and maximize the cumulative reward at each step, by choosing an action. That is why the most common quantity of interest in RL is the discounted cumulative reward :

$$R(\tau) = \sum_{t=0}^T \gamma^t r_t, \quad (4)$$

where T is the final time, and $\gamma \in [0, 1]$ is the discount factor that weights the importance of rewards according to their distance to the present (the agent being short-sighted in the limit where $\gamma \rightarrow 0$, and far-sighted in the limit where $\gamma \rightarrow 1$).

There exist two main types of RL algorithms : model-based methods either have access to the environment and therefore know the probability distribution of the states they end up in, or try to build an approximation of it ; model-free methods don't try to understand the environment, but only communicate with it and try to find the best actions to take, these methods are prominent in the DRL community.

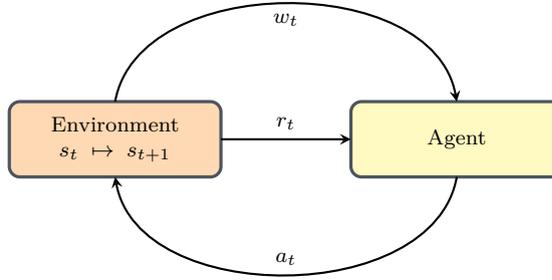


Figure 2: RL agent and its interactions with its environment.

101 Inside the model-free family of algorithms are also two main techniques : value-based methods
 102 that learn to predict the future rewards of a set of actions in order to pick the best one, and policy-
 103 based methods that keep in memory a policy that maps states to actions and learn to obtain the
 104 best rewards by modifying this policy. Many DRL algorithms in the community, including PPO,
 105 the one used in this paper, use gradient ascent to optimize a parameterized policy with respect to
 106 the expected return and therefore belong to the family of policy gradient methods. For a more
 107 thorough introduction to the taxonomy of RL methods (together with their respective pros and
 108 cons) please refer to Ref. [32].

109 3.3. From policy methods to Proximal policy optimization

110 This section is intended for non-expert readers and provides an overview of the basic principles
 111 and prerequisites of the Policy Gradient Method and the various steps taken to improve it.
 112

113 - *Policy methods.* A policy method maximizes the expected discounted cumulative reward of a
 114 decision policy π mapping states to actions. It doesn't use a value function as explained before,
 115 but a probability distribution to determine which actions are best at any given state. Policies being
 116 often stochastic, the following notations are introduced:

- 117 • $\pi(s, a)$ is the probability of taking action a in state s under policy π ,
- 118 • $Q^\pi(s, a)$ is the expected value of the discounted cumulative reward after taking action a in
 119 state s (also termed state-action value function or Q-function)

$$Q^\pi(s, a) = \mathbb{E}_\pi [R(\tau)|s, a], \quad (5)$$

120 where \mathbb{E}_π is the expected value \mathbb{E} under policy π .

- 121 • $V^\pi(s)$ is the expected value of the discounted cumulative reward in state s (also termed value
 122 function or V-function)

$$V^\pi(s) = \mathbb{E}_\pi [R(\tau)|s]. \quad (6)$$

123 The V and Q functions are thus related such that

$$V^\pi(s) = \sum_a \pi(s, a) Q^\pi(s, a), \quad (7)$$

124 so $V^\pi(s)$ can also be understood as the probability-weighted average of discounted cumulative
 125 rewards over all possible actions in state s .

126 - *Policy gradient methods.* A policy gradient method aims at optimizing a parameterized policy
 127 π_θ , where θ denotes the free parameters whose values can be learnt from data (as opposed to the
 128 hyper parameters). In practice, one defines an objective function based on the expected discounted
 129 cumulative reward

$$J(\theta) = \mathbb{E}_{\pi_\theta} [R(\tau)], \quad (8)$$

130 and looks for θ^* maximizing $J(\theta)$:

$$\theta^* = \arg \max_{\theta} \mathbb{E}_{\pi_\theta} [R(\tau)]. \quad (9)$$

131 One can try to do this by estimating the policy gradient $\nabla_\theta J(\theta)$ and using a gradient ascent
 132 algorithm. This is certainly a difficult task as one is seeking a gradient that depends on the policy
 133 parameters, but also on the whole space of state-action pairs, in a context where the effects of
 134 policy changes on the state probability distribution are unknown (since modifying the policy will
 135 most likely modify the probability distribution over the set of visited states). One commonly used
 136 estimator, derived in [32] using the log-probability trick, reads

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^T \nabla_\theta \log(\pi_\theta(s_t, a_t)) R(\tau) \right] \sim \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^T \nabla_\theta \log(\pi_\theta(s_t, a_t)) \hat{A}^\pi(s_t, a_t) \right], \quad (10)$$

137 where \hat{A}^π is some biased estimate (here its normalization to zero mean and unit variance) of the
 138 advantage function

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s), \quad (11)$$

139 that measures the improvement (if $A^\pi > 0$, otherwise the lack thereof) associated with taking action
 140 a in state s (Q-function) compared to taking the average over all possible actions (V-function).
 141 This is possible because the V-function doesn't depend on the action, and therefore doesn't change
 142 the expected value, but it shows experimentally that it also reduces the variance and speeds up the
 143 learning. When the policy π_θ is represented by a neural network (in which case θ simply represents
 144 the network parameters), we tend to estimate the policy loss

$$L(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^T \log(\pi_\theta(a_t|s_t)) \hat{A}^\pi(s_t, a_t) \right], \quad (12)$$

145 whose gradient is equal to the (approximated) policy gradient (10) and is computed with respect
 146 to each parameter of the neural network by using the chain rule at each layer with the back-
 147 propagation algorithm [33].

148
 149 - *Trust regions.* The learning rate of the policy gradient methods, i.e. the size of the steps taken at
 150 each learning iteration, has a large impact on their performances. Too small, and the learning will
 151 never end, too large, and it will be difficult to get out of degenerate regions where the gradient is
 152 already high or noisy. Fine-tuning the learning rate could be a solution, but it asks sometimes for
 153 too much work finding the right balance. One way to stay in the range of improvement is to define
 154 a maximum distance between the new policy and the old one, this way, even when the gradient
 155 becomes too high, the trust region clips the distance and avoid the aforementioned issues. We
 156 will not dwell on the intricate details of the many algorithms developed to solve such trust region
 157 optimization problems, e.g., natural policy gradient (NPG [34]), or trust region policy optimiza-
 158 tion (TRPO [35]). Suffice it to say that they use the MinMax algorithm to maximize iteratively
 159 a surrogate policy loss (i.e. a lower bound approximating locally the actual loss at the current
 160 policy), but are difficult to implement and can be computationally expensive, as they rely on an
 161 estimate of the second-order gradient of the policy log probability.

162

163 - *Proximal policy optimization.* Proximal policy optimization (PPO) is a similar approach to TRPO
 164 (on which it is based) but with a simpler heuristic that uses a probability ratio between the two
 165 policies to maximize improvement without the risk of performance collapse [25]. The focus here is
 166 on the PPO-clip algorithm¹ that optimizes the surrogate loss

$$L(\theta) = \mathbb{E}_{\pi_\theta} \left[\min \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}, g(\epsilon, \hat{A}^\pi(s, a)) \right) \hat{A}^\pi(s, a) \right], \quad (13)$$

167 where

$$g(\epsilon, A) = \begin{cases} 1 + \epsilon & A \geq 0, \\ 1 - \epsilon & A < 0, \end{cases} \quad (14)$$

168 and $\epsilon \in [0.1, 0.3]$ is the clipping range, a small hyper parameter defining how far away the new
 169 policy is allowed to go from the old. Its range is adapted from the paper on Proximal Policy
 170 Optimization [36] and was confirmed in our own implementation. The general picture is that
 171 a positive (resp. negative) advantage increases (resp. decreases) the probability of taking action
 172 a in state s , but always by a proportion smaller than ϵ , otherwise the min kicks in [13] and its
 173 argument hits a ceiling of $1 + \epsilon$ (resp. a floor of $1 - \epsilon$). This prevents stepping too far away from
 174 the current policy, and ensures that the new policy will behave similarly but hopefully in a better
 175 way.

176 3.4. Single-step PPO

177 We now come to single-step PPO, a “degenerate” version of PPO introduced in [26] and intended
 178 for situations where the optimal policy to be learnt by the neural network is state-independent,
 179 i.e. $\pi_\theta(a, s) = \pi_\theta(a)$, as is notably the case in optimization and open-loop control problems
 180 (closed-loop control problems conversely require state-dependent policies for which standard PPO
 181 is best suited). The main difference between standard and single-step PPO can be summed up as
 182 follows: where standard PPO seeks the optimal set of parameters θ^* leading to the largest possible
 183 cumulative reward over one episode, single-step PPO seeks the optimal parameters θ^* such that
 184 $a^* = \pi_{\theta^*}(s_0)$, where s_0 is some input state (usually a constant vector of zeros) consistently fed
 185 to the agent for the optimal policy to eventually embody the transformation from s_0 to a^* . The
 186 agent initially implements a random initial policy determined by the free parameters θ_0 , after
 187 which it gets only one attempt per learning episode at finding the optimal (i.e., it interacts with
 188 the environment only once per episode). This is illustrated in figure 3 showing the agent draw a
 189 population of actions a_t from the current policy, and being returned incentives from the associated
 190 rewards to update the free parameters for the next population of actions $a_{t+1} = \pi_{\theta_{t+1}}(s_0)$ to yield
 191 larger rewards.

192 In practice, the agent outputs a policy parameterized by the mean and variance of the proba-
 193 bility density function of a d -dimensional multivariate normal distribution, with d the dimension
 194 of the action required by the environment. Actions drawn in $[-1, 1]^d$ are then mapped into rele-
 195 vant physical ranges, a step deferred to the environment as being problem-specific. The resolution
 196 essentially follows the process described in section 3.3, only a normalized averaged reward substi-
 197 tutes for the advantage function. This is because classical PPO is actor-critic, i.e., it improves the
 198 learning performance by updating two different networks, a first one called actor that controls the
 199 actions taken by the agent, and a second one called critic, that learns to estimate the advantage
 200 from the value function as

$$A(s_t, a_t) = r_t + \gamma V(s_{t+1}) - V(s_t). \quad (15)$$

201 In single-step PPO, the trajectory consists of a single state-action pair, so the discount factor can
 202 be set to $\gamma = 1$ with no loss of generality. In return, the advantage reduces to the whitened reward
 203 since the two rightmost terms cancel each other out in [15]. This means that the approach can do
 204 without the value-function evaluations of the critic network, i.e., it is not actually actor-critic.

¹There is also a PPO-Penalty variant which uses a penalization on the average Kullback–Leibler divergence between the current and new policies, but PPO-clip performs better in practice.

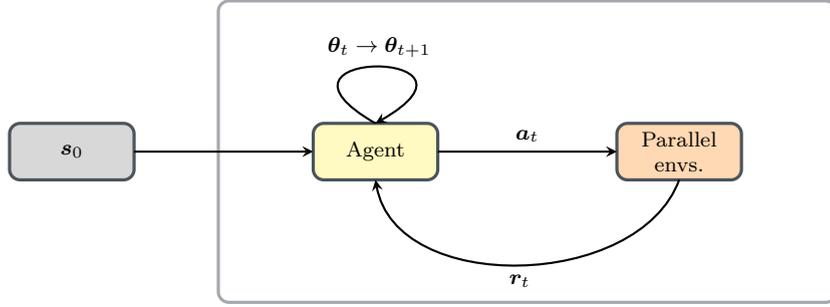


Figure 3: Action loop for single-step PPO. At each episode, the input state s_0 is provided to the agent, which in turn provides n actions to n parallel environments. The latter return n rewards, that evaluate the quality of each action taken. Once all the rewards are collected, an update of the agent parameters is made using the PPO loss (13).

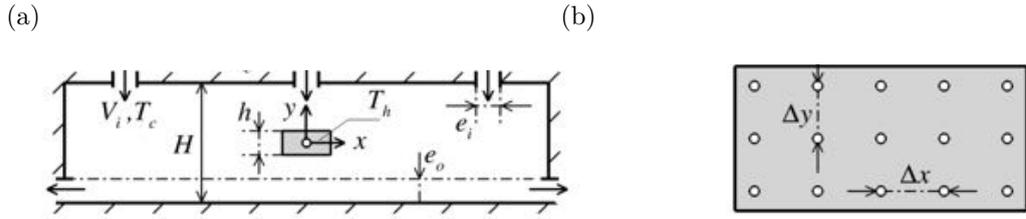


Figure 4: (a) Schematic of the 2D forced convection set-up. (b) Sensors positions in the solid domain.

205 4. Control of forced convection heating in a 2D open cavity

206 4.1. Case description

207 This test case is based on the second test case of this paper [24] with a few twists. Here we
 208 address the control of conjugate heat transfer for the heating of a piece by injection of a hot fluid
 209 in the chamber. We use a Cartesian coordinate system with origin at the center of the chamber.
 210 The solid has a rectangular shape with height h and aspect ratio 2:1, and is initially at the cold
 211 temperature T_c . It can be fixed or move around the chamber according to the parameters we wish to
 212 control. Its density, thermal conductivity and heat capacity can take two values, depending on the
 213 type of solid we are trying to emulate in order to compare the two : brick or steel. The chamber
 214 itself has a rectangular shape with height H and aspect ratio 4:1, and its walls are isothermal at
 215 temperature T_w . The north wall of the chamber has three identical inlets of width e , each of which
 216 models the exit plane of an injector blowing hot air with constant temperature T_h and velocities
 217 $V_i \in [0.01, 0.99]$ subjected to

$$\sum_{i=1}^3 V_i = 1, \quad (16)$$

218 to emulate a constant input to the chamber. The fluid is released through two identical outlets on
 219 each side of the chamber with height e_0 , and positioned against the south wall.

220 In the absence of buoyancy, temperature evolves as a passive scalar to the Navier–Stokes equations.
 221 All parameters named above are provided in [1], along with the material properties used to model
 222 the composite fluid, that yield fluid values of the Reynolds and Prandtl numbers

$$\text{Re} = \frac{\rho e \max_{i \in \{1,2,3\}} V_i}{\mu} \in [67, 200], \quad \text{Pr} = \frac{c_p \mu}{\lambda} = 2. \quad (17)$$

223 Note the very high value of the solid to fluid viscosity ratio, meant to ensure that the velocity
 224 inside the solid is zero and that the no-slip condition on the boundary is satisfied. Thus, only
 225 pure conduction occurs in the solid. The governing equations are solved with no-slip isothermal
 226 conditions $\mathbf{u} = \mathbf{0}$ and $T = T_w$ on $\partial\Omega$, except at the inlets where $\mathbf{u} = -V_i \mathbf{e}_y$ and $T = T_h$, and
 227 at the exhausts where a zero-pressure condition is imposed : $p = \partial_x u = \partial_x T = 0$. No thermal
 228 condition is imposed at the interface, where heat exchange is implicitly driven by the difference

H	h	e	T_w	T_c	T_h	μ	ρ	λ	c_p	
						0.001	1	0.5	1000	Fluid
1	0.2	0.2	10	10	300	1000	1800	1	600	Brick
						1000	7800	50	500	Steel

Table 1: Geometric and numerical parameters used in the 2-D forced convection setup. All values in SI units, with the exception of temperature given in Celsius.

229 in the individual material properties, as intended in the Immersed Volume Method. The mesh on
 230 which the computation is performed is defined at the beginning only around the interface to ensure
 231 its good definition, then remeshing is performed every 5 time steps based on velocity gradients
 232 to also capture accurate velocity profiles. We aim at 20000 elements, with $h_{min} = 0.0001$. More
 233 about our remeshing technique here [37].

234 4.2. Control

235 The quantities subjected to optimization are the three inflow velocities $V_{i \in \{1,2,3\}}$, plus the
 236 insertion angle of the piece being heated, and the position of its center of mass, hence six control
 237 parameters (three for the inflow distribution, one for the angle and two for the position). We could
 238 have used two parameters to control the inflow distribution since the last one is constrained by the
 239 total inflow, but in order to avoid asymmetry in the learning process we decided to control each
 240 of the injectors in the same way. In practice, each injector is given a value between 0.01 and 0.99
 241 which is then scaled in order to obtain their speed while following [16].

242 Just like in [24], we distribute 15 probes uniformly in the workpiece to compute the reward used
 243 by the DRL algorithm. The probes are arranged in an array of $n_x = 5$ columns and $n_y = 3$ rows
 244 with resolutions $\Delta x = 0.09$ and $\Delta y = 0.075$, respectively; see figure 4(b). The following formula
 245 gives an estimate of the tangential heat flux by averaging the norm of the temperature gradient
 246 across rows and columns respectively :

$$\langle \|\nabla_{\parallel} T\| \rangle_i = \frac{2}{n_y - 1} \left| \sum_{j \neq 0} \text{sgn}(j) \|\nabla T\|_{ij} \right|, \quad \langle \|\nabla_{\parallel} T\| \rangle_j = \frac{2}{n_x - 1} \left| \sum_{i \neq 0} \text{sgn}(i) \|\nabla T\|_{ij} \right|, \quad (18)$$

247 where subscripts i , j and ij denote quantities evaluated at $x = i\Delta x$, $y = j\Delta y$ and $(x, y) =$
 248 $(i\Delta x, j\Delta y)$, respectively, and symmetrical numbering is used for the center probe to sit at the
 249 intersection of the zero-th column and row. The reward $r_t = -\langle \|\nabla_{\parallel} T\| \rangle$ fed to the DRL agent is
 250 given by the average of the quantities calculated before

$$r_t = -\frac{1}{n_x + n_y} \sum_{i,j} \langle \|\nabla_{\parallel} T\| \rangle_i + \langle \|\nabla_{\parallel} T\| \rangle_j, \quad (19)$$

251 which especially yields $r_t = 0$ for a perfectly homogeneous heating.

252 A second reward is also tested in this paper to assess the feasibility of controlling both the
 253 homogeneity and efficiency of the furnace. This is expressed as

$$\psi_t(w_1, w_2) = \frac{w_1}{n_x n_y} \sum_{i,j} T_{ij} + w_2 r_t, \quad (20)$$

254 where the first term is the right-hand side measures the solid temperature averaged across all
 255 sensors, and $w_{1,2}$ are scalar-valued factors weighing the priority given to each objective. In practice,
 256 a single point concurrently minimizing both objectives usually does not exist. The optimal solutions
 257 are thus to be understood as Pareto-efficient solutions [38] that best manage trade-offs between
 258 the two criteria, in the sense that further optimizing one objective decreases the performance
 259 of the other one (after which the final decision is made by the practitioner based on subjective
 260 preferences).

261 The agent is a fully-connected network with two hidden layers, each holding 2 neurons. The
 262 resolution process uses 8 environments and 2 steps mini-batches to update the network for 32
 263 epochs, with learning rate set to 5×10^{-3} , and PBO loss clipping range to $\epsilon = 0.3$.

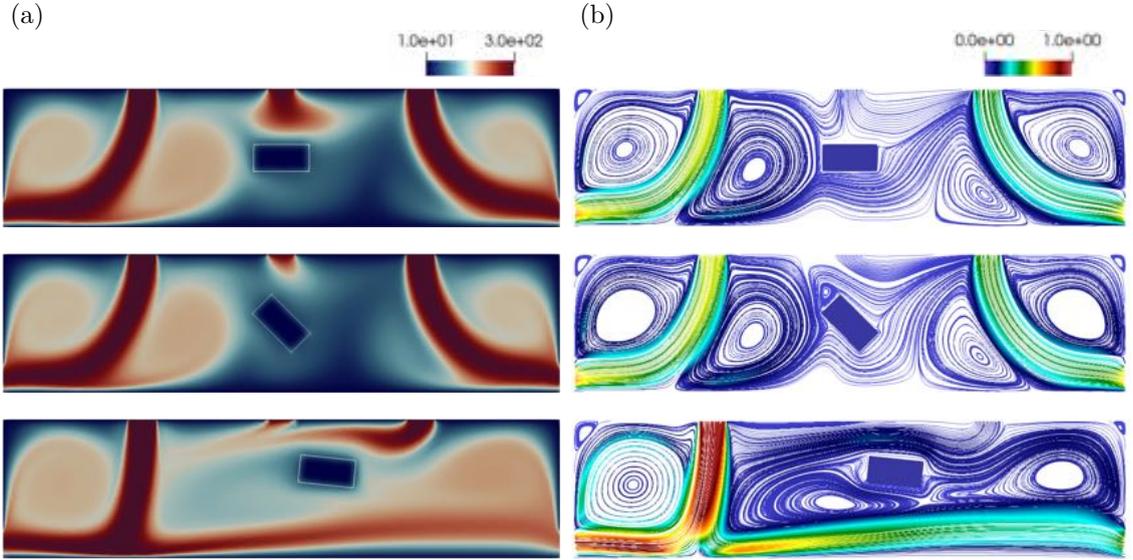


Figure 5: Control of heating homogeneity for the brick-like workpiece. (a) Optimal temperature distribution found by the DRL algorithm by controlling (from top to bottom) : the inflow velocities under constant workpiece angle and position, the inflow velocities and workpiece angle under constant position, and the inflow velocities, workpiece angle and position. (b) Corresponding streamlines colored by the magnitude of velocity.

264 4.3. Results

265 4.3.1. Control of heating homogeneity

266 We evaluate first the performance of the algorithm in several scenarios of increasing complexity,
 267 using the homogeneity-based reward r_t . Two different workpieces are used, one that has the
 268 properties of a brick-like material, and one that has the properties of a steel-like material. For each
 269 workpiece, three different cases are considered, in which the DRL agent is tasked with optimizing
 270 either the inflow velocities (under constant workpiece angle and position), or the inflow velocities
 271 and the workpiece angle (under constant position), or the inflow velocities, angle and position (the
 272 most general case).

273 For each case, 150 episodes have been run (1200 simulations), each of which performs 2000
 274 iterations with time step $\Delta t = 0.1$ (hence a heating time of 200), starting from an initial condition
 275 consisting of zero velocity and uniform temperature (except in the solid domain), and using the
 276 level set, velocity and temperature as multiple-component criterion to adapt the mesh (initially
 277 pre-adapted using the sole level set) every 5 time steps under the constraint of a fixed number of
 278 elements $n_{el} = 15000$. This represents 1200 simulations, each of which is performed on 8 cores and
 279 lasts 10mn, hence 200h of total CPU cost. We can clearly see in figures 5 and 6 the flow patterns
 280 that develop when the blown fluid travels through the cavity. Moreover, it clearly depends on the
 281 inflow distribution and position of the piece, and features complex rebound phenomena (either
 282 fluid/solid, when a jet impinges on the workpiece, or fluid/fluid, when a deflected jet meets the
 283 crossflow of another jet), leading to the formation of multiple recirculations varying in number,
 284 position and size.

285 The results of the various optimization scenarios are shown in figure 5 for the brick-like material
 286 and figure 6 for the steel-like material. The use of the reward r_t has a great effect on the aspect
 287 of these results in any control configuration. Even though they are all different, we can spot some
 288 common features : the workpiece is kept in colder areas of the chamber, which allows for lower
 289 temperature gradients as prescribed by the reward, and it is also generally well surrounded by
 290 streamlines that ensure symmetry in the heating. In the simplest case where the agent has to
 291 control only the inflow velocities, it has no problem finding the best solution possible, which is true
 292 also when we add the workpiece angle (not shown here for conciseness), and they both present the
 293 features we mentioned.

294 When tasked with simultaneously optimizing all six parameters (inflow velocities, angle and
 295 position), we show in figure 7 that the algorithm learns quite fast up to episode 60, but pursues

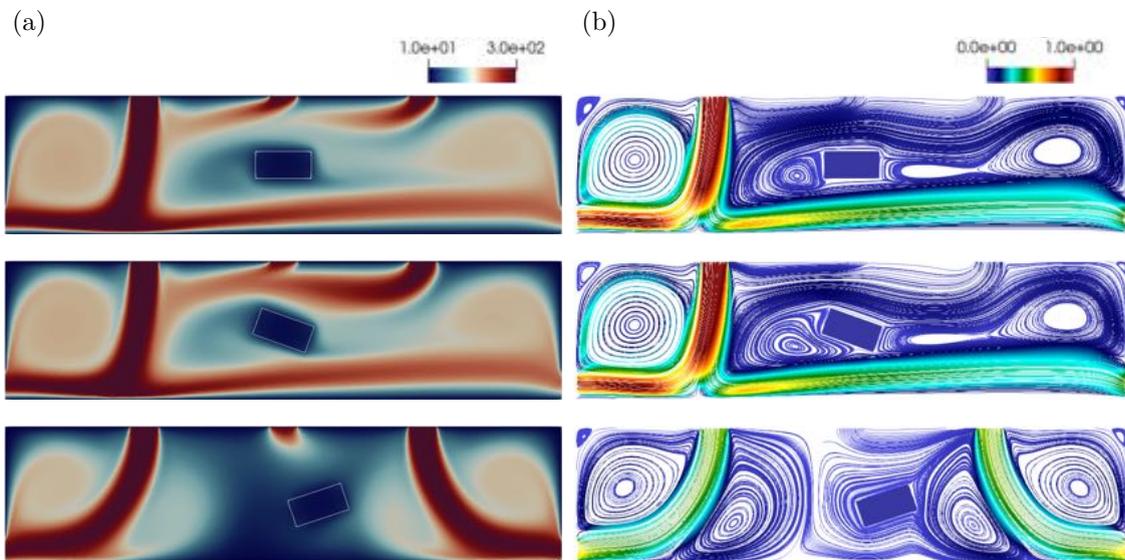


Figure 6: Control of heating homogeneity for the steel-like workpiece. (a) Optimal temperature distribution found by the DRL algorithm by controlling (from top to bottom) : the inflow velocities under constant workpiece angle and position, the inflow velocities and workpiece angle under constant position, and the inflow velocities, workpiece angle and position. (b) Corresponding streamlines colored by the magnitude of velocity.

296 the iteration to handle the position on the y axis and the angle, as it restarts an exploration phase
 297 around episode 85, which allows to reach even better rewards in the end (see also figure 8 for
 298 an illustration of temperature distributions at the final step randomly sampled over the course of
 299 optimization). Indeed, with a larger space to explore, some parameters may be optimized more
 300 easily than others, for example if they have a greater impact on the reward. In this case, this may
 301 be a good idea to optimize the last parameters individually, since the first reward chosen may not
 302 be the best fitted to their optimization. The reward obtained from the original simulation (centered
 303 piece, same speed at each inlet) has a mean value of -0.450 , figure 7 indicates that PPO converged
 304 around a reward of -0.050 which yields an improvement of 89% of the temperature homogeneity.

305 4.3.2. Control of heating efficiency

306 As explained in 4.2, we introduce here a second reward ψ_{t,w_1,w_2} to assess the feasibility of con-
 307 trolling the efficiency of heating of the piece while keeping the gradient homogeneous. The PPO
 308 algorithm should consider a new solution space in which minima are decided by both the tem-
 309 perature and the gradient. This should lead to different solutions than the ones before, especially
 310 concerning the average of the temperature values at the probes.

311 Only the steel-like workpiece is considered here (as it allows for a better conduction and there-
 312 fore larger temperature differences) under the third optimization scenario, *i.e.*, inflow velocities,
 313 workpiece angle and position as free parameters). We tested two sets of weight, $(w_1, w_2) = (1, 1)$
 314 and $(10, 1)$, with the second one giving more priority to the averaged temperature component,
 315 and compare in figure 9 the obtained average temperature at the final time step, to those ob-
 316 tained under the three scenario presented earlier for pure homogeneous control (that corresponds
 317 to $(w_1, w_2) = (0, 1)$). For the simplest homogeneous control cases presented in figures 9(a) and
 318 (b), the temperature is constrained by the position of the workpiece, and the algorithm quickly
 319 converges. Adding in the workpiece position as free parameter increases the complexity, (the pos-
 320 sibility to move the workpiece anywhere in the chamber yields much higher variance in the space
 321 of achievable temperature distributions). In return, the average temperature in figure 9(c) drops
 322 to a lower value value slightly above 26° , to be understood as an indirect consequence of the op-
 323 timization of the homogeneity reward r_t . The scenario shown in figure 9(d) corresponds to the
 324 optimization of the compound reward ψ_t using all six control parameters $(w_1, w_2) = (1, 1)$, for
 325 which the temperature again drops, similar to the previous case. Finally, by using the reward
 326 $\psi_{t,10,1}$ with $(w_1, w_2) = (10, 1)$, the algorithm reaches a temperature of 26.5° . Consistently, the
 327 optimal temperature distributions for this two cases are somewhat similar to those obtained by

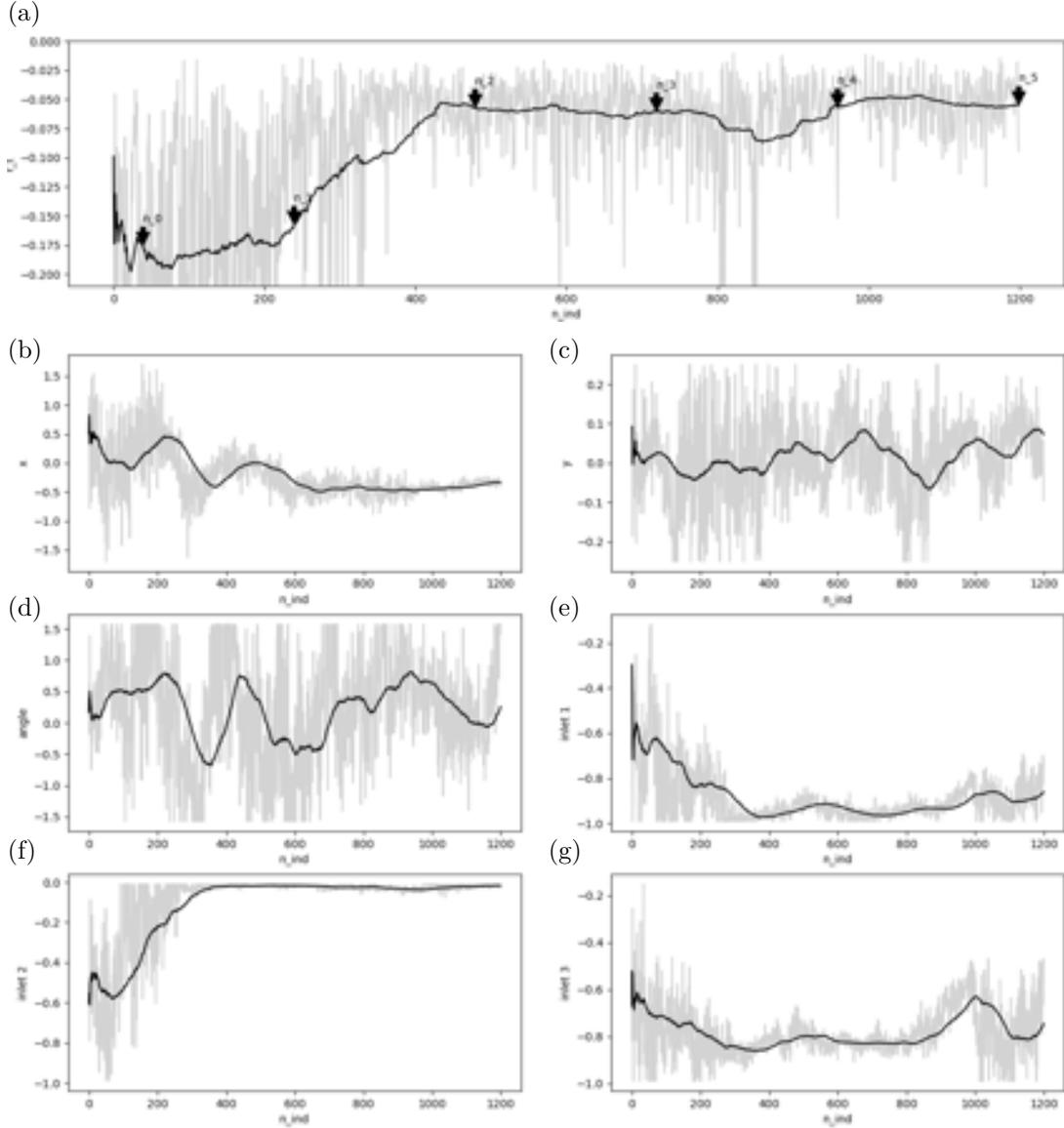


Figure 7: Control of heating homogeneity for the steel-like workpiece using the inflow velocities, workpiece angle and position as free parameters, corresponding to the scenario at the bottom of figure 6. Evolution per episode of the (a) reward, and (b-g) control parameters. Black curves are the moving averages.

328 controlling the heating homogeneity, but with a workpiece that is closer to the heat sources as the
 329 value of w_1 in the reward ψ_t increases, to give more priority to lowering the averaged temperature.

330 4.4. Discussion

331 Whether it be for the brick-like or the steel-like workpiece, the algorithm finds, in each scenario,
 332 a relevant local minimum that satisfies the conditions we imposed on it. One point worth noticing
 333 is the wide variety of solutions the algorithm comes up with, best illustrated by comparing figures
 334 5, 6 and 10. This may be because the complexity of this case gives room for a lot of equivalent
 335 solutions, and the algorithm struggles to find the global minimum and always ends up in local
 336 minima. Numerical approximations can also create noise and give an information too imprecise to
 337 be processed by the PPO method, especially since the reward is calculated from point-wise data
 338 interpolated from the simulation (similar to experimental measurements). By running the same
 339 simulation 1000 times, we noticed a fluctuation in the reward, with estimated relative standard
 340 deviation by 4.1%.

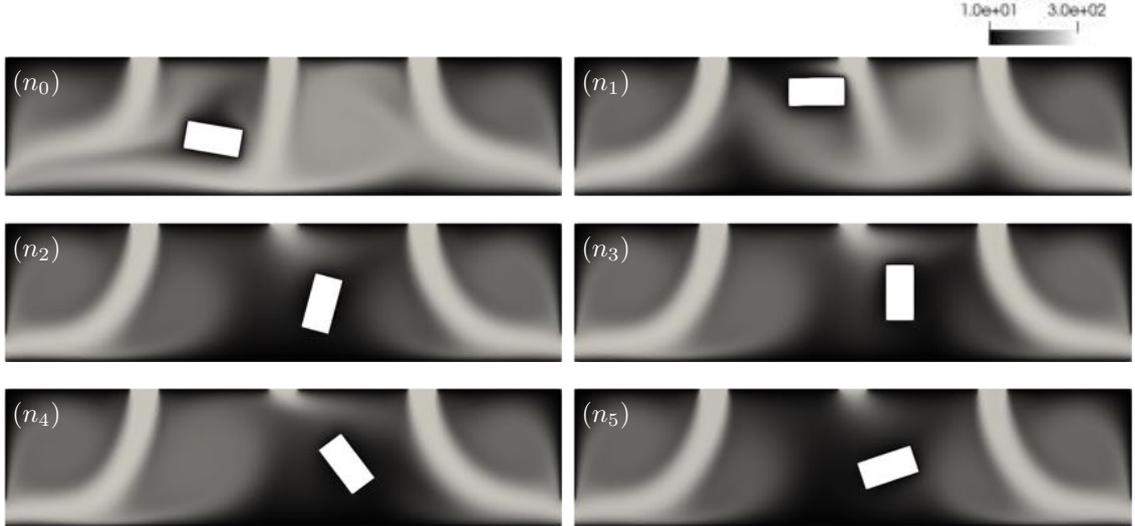


Figure 8: Control of heating homogeneity for the steel-like workpiece using the inflow velocities, workpiece angle and position as free parameters, corresponding to the scenario at the bottom of figure 6. Temperature distribution at the final time step for randomly sampled episodes marked by the arrows in figure 7(a).

341 This has been confirmed by running 8 times the same learning experiment (steel workpiece,
342 homogeneous reward r_t , control of the inflow velocities, workpiece angle and position), for which
343 the algorithm found local minima more or less sensitive to perturbations, and the relative standard
344 deviation computed for the moving average over the 50 latest values is 9.9%. As interpreted before,
345 the algorithm seems to struggle finding a global maximum of our reward function r_t , likely because
346 there exists a sensitivity to noise and system uncertainty. There are a few possible reactions to
347 that. Firstly, it could be useful to improve the balance between exploration and exploitation,
348 using improved search distributions to effectively encourage the policy to explore more on the
349 potential valuable actions, no matter whether they were preferred by the previous policies or not
350 (for instance, using the recently introduced PBO algorithm [39], that uses three separate neural
351 networks to learn the mean, variance and correlation parameters of a multivariate normal search
352 distribution, while single-step PPO updates the mean and variance, the same for all variables,
353 from a single neural network). Another possibility is to fine-tune the architecture of the neural
354 network. As the number of parameters to control increases, the number of neurons that model
355 the policy should increase too, but there is no guideline for that. Finally, the reward itself could
356 be changed, along with the constraints on the parameters to control. These aspects have a great
357 impact on the shape of the solution space, and therefore on the ease for the algorithm to find a
358 global maximum. In real world applications, a lot of constraints can come into consideration when
359 choosing the reward, or the parameters to operate on. Finding the right way to communicate our
360 needs to the algorithm is a whole topic in itself. We leave the exploration of these reactions to
361 future works.

362 5. Single-step control of a 3D serpentine heater

363 5.1. Case description

364 We propose in this test case to apply the same DRL-CFD framework to a three-dimensional
365 case of industrial interest: the serpentine heater. It consists of a 3D simulation of two fluids, a
366 liquid which is cold at first, and the hot gas that is distributed inside the chamber, and should come
367 out colder due to the transfer of heat from the gas to the liquid (this resembles a heat exchanger
368 but with a configuration closer to a gas burner).

369 The control objective here is to find the best flow distribution between an array of gas burners
370 to heat a liquid in a pipe. We use Cartesian coordinate system with origin at the center of the
371 chamber. The chamber is a simple parallelepiped with size H on x and z axes and h on the y axis.
372 The pipe is made from the extrusion of a circle of radius R and a center at $(-H/2, 0, -H/4)$, with
373 three 180 degrees bends around the y axis. It has two planes of symmetry Oxz and Oxy . The
374 longer straight part of the pipe has a length of L , the second straight part has a length of l and

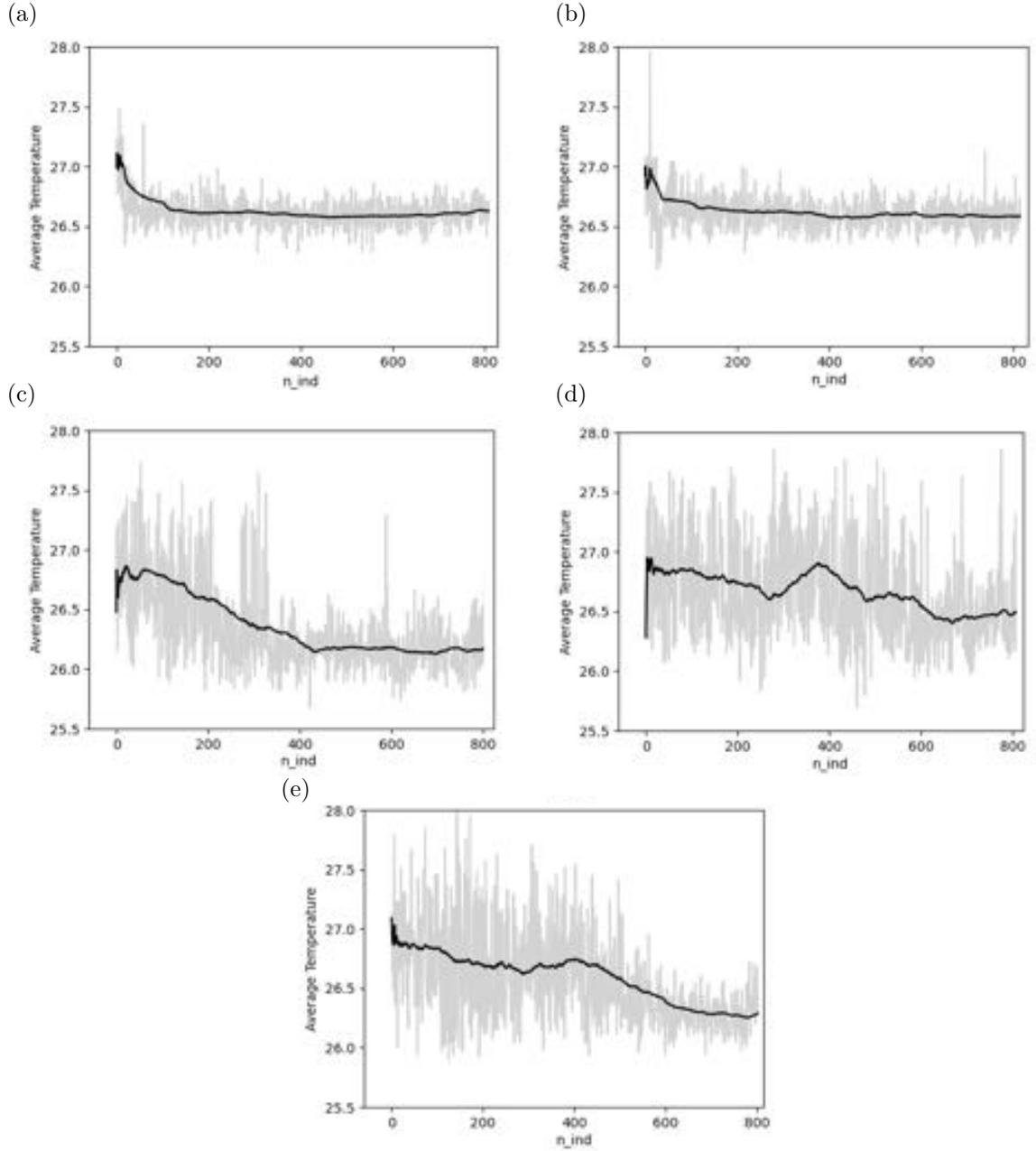


Figure 9: Evolution per episode of the final temperature averaged across all sensor positions in the steel-like workpiece, using (a-c) the homogeneity reward and (d-e) the compound homogeneity/efficiency reward with (d) $(w_1, w_2) = (1, 1)$ and (e) $(w_1, w_2) = (10, 1)$. The free parameters subjected to optimization are (a) the inflow velocities under constant workpiece angle and position, (b) the inflow velocities and workpiece angle under constant position, and (c-e) the inflow velocities, workpiece angle and position.

375 the rest of the pipe can be deduced from symmetry with the condition that all three bends have
 376 exactly the same radius. The hot gas comes out of 6 circular inlets of radius r , 4 on the y^- side
 377 and two on the y^+ side. They are disposed in an array aligned on the center of the pipe at each of
 378 its bends, see figure [12](#) (a) et (c). Since this case has no moving parts, remeshing is not needed,
 379 hence a unique mesh is used, as seen in figure [12](#) (d). It is composed of 35021 nodes and 197949
 380 tetrahedral elements. The elements aspect ratio is shown in the bar chart [13](#).

381 The fluid in the pipe comes in through the lower hole with a parabolic speed profile correspond-
 382 ing to a flow rate D , and a temperature T_c . At $t = 0$ the temperature in the chamber is T_c , and hot
 383 gas comes through the 6 inlets at temperature T_h , with speed $V_i \in [0.1, 0.9]$. In the same manner

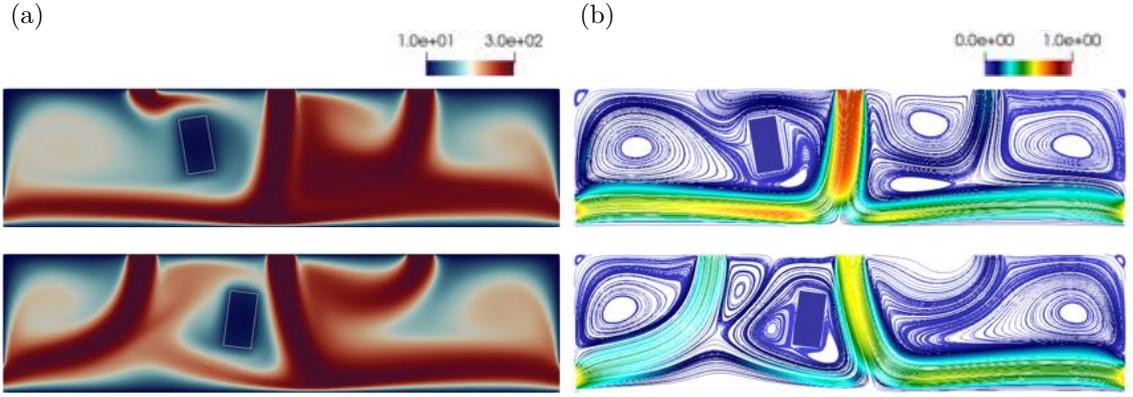


Figure 10: Control of efficiency for the steel-like workpiece. (a) Optimal temperature distribution found by the DRL algorithm by controlling the inflow velocities, workpiece angle and position, using reward ψ_t with weights (w_1, w_2) set to (from top to bottom) $(1, 1)$ and $(10, 1)$. (b) Corresponding streamlines colored by the magnitude of velocity.

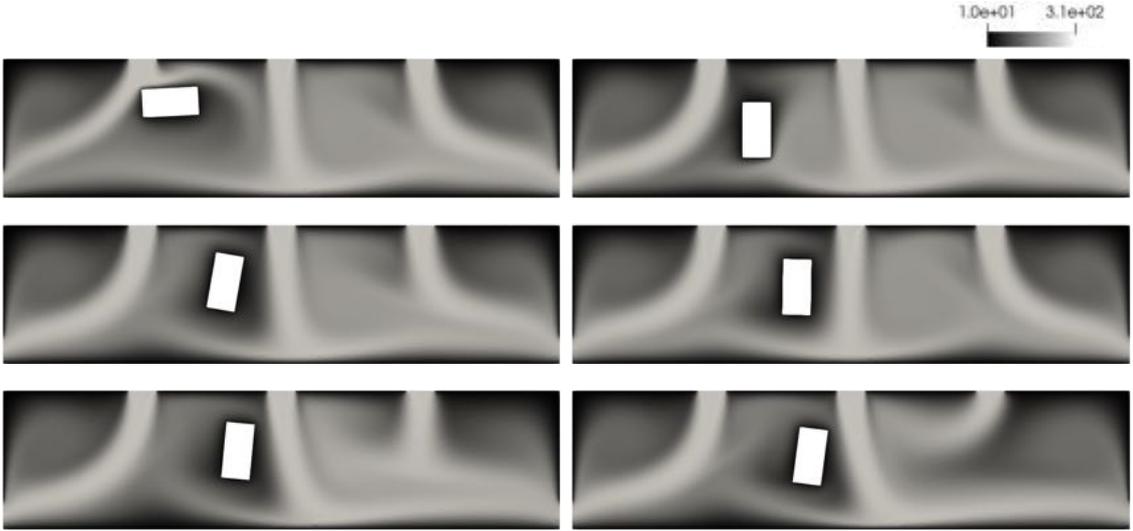


Figure 11: Temperature distribution at the final time step for randomly sampled episodes of heating efficiency control for the steel-like workpiece. The free parameters subjected to optimization are the inflow velocities, workpiece angle and position, corresponding to the scenario at the bottom of figure [10](#).

384 as for the previous case, we emulate a constant input by imposing

$$\sum_{i=1}^6 V_i = 1. \quad (21)$$

385 The exhaust is positioned in the center of the top face of the chamber. It has a rectangular shape
 386 and is half the size of the top face. We consider in this case that there is no solid material separating
 387 the inside of the pipe from the gas. Only a no slip and isothermal condition is applied on the walls
 388 of the chamber (with the exception of the inlets and exhausts), with $T = T_w$. A no slip condition
 389 is also applied on the 2D interface representing the pipe. At the exhausts a zero-pressure condition
 390 is imposed : $p = \partial_x u = \partial_x T = 0$. No thermal condition is imposed at the interface, where heat
 391 exchange is implicitly driven by the difference in the individual material properties. Again, in the
 392 absence of buoyancy, the temperature evolves as a passive scalar for the Navier–Stokes equations.
 393 This allows us to solve the Navier–Stokes equations separately for the gas and the fluid in the
 394 pipe and then find the temperature field in the whole domain. Using the parameters for the liquid
 395 and the gas provided in Table [2](#), the Reynolds and Prandtl numbers in the hot gas domain are
 396 estimated to be $\text{Re} \in [5, 30]$ and $\text{Pr} = 20$, and $\text{Re} \in [320]$ and $\text{Pr} = 400$ in the liquid.

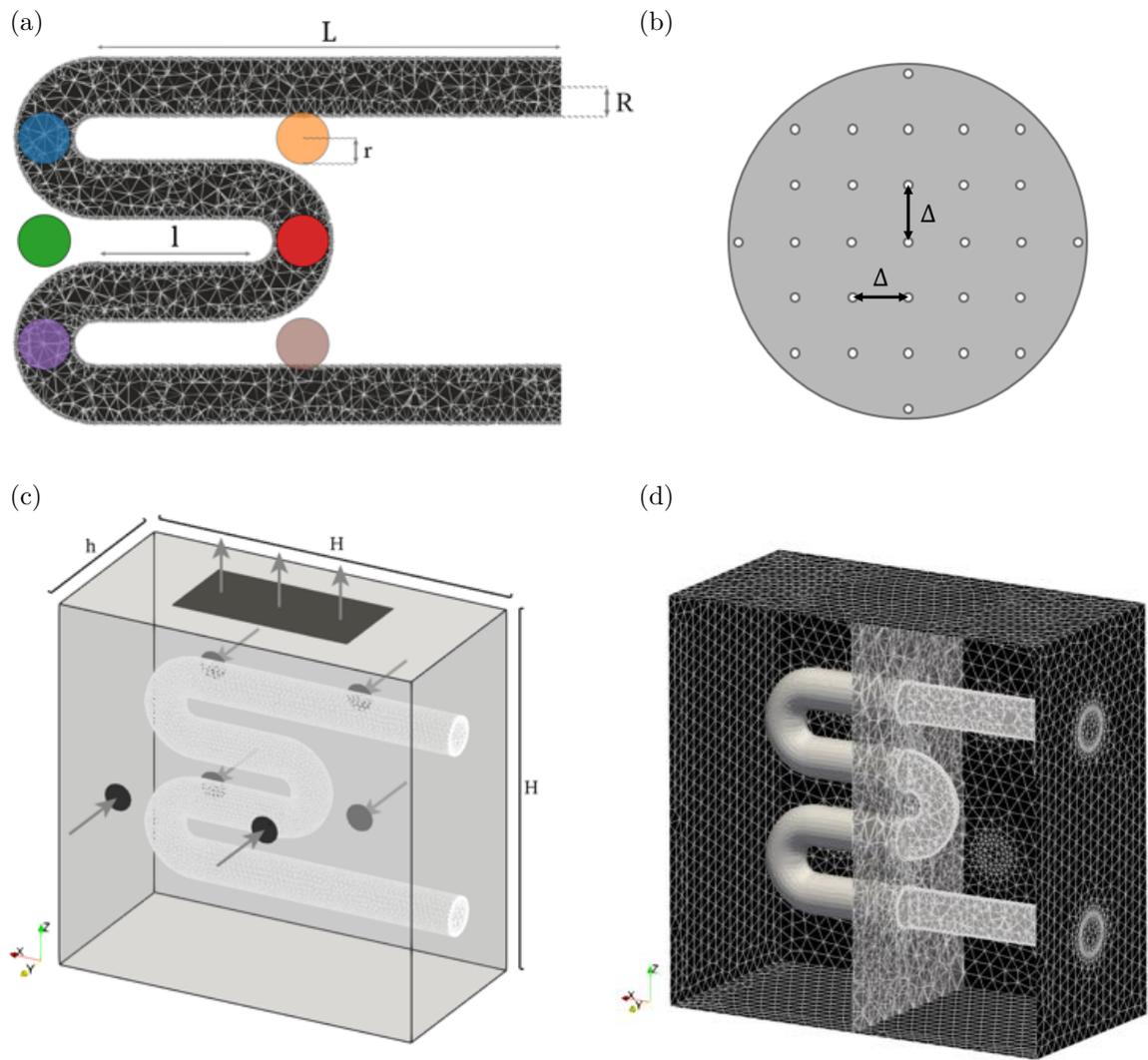


Figure 12: (a) Schematic of the pipe with positions of the gas inlets projected along the y axis. Fully opaque circles are inlets in front of the pipe, i.e. on the y^+ side of the chamber, and transparent circles are inlets on the other side. (b) Position of the sensors on the outlet of the pipe. (c) Schematic of the whole setup with arrows representing the direction of the gas flow. (d) Representation of the mesh used.

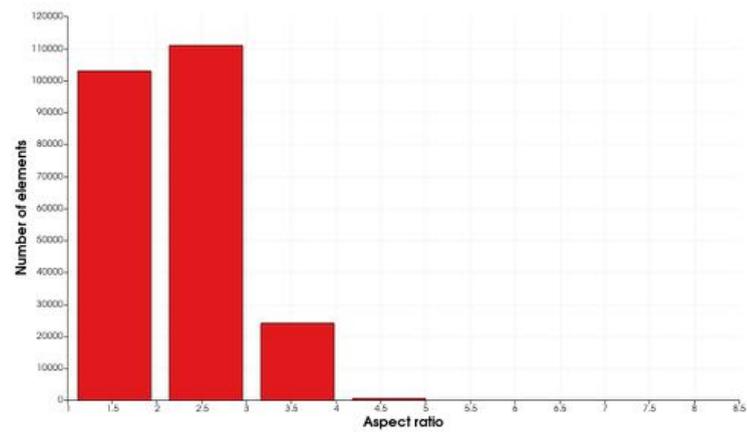


Figure 13: Bar chart of the aspect ratio of tetrahedral elements.

H	h	R	r	L	l	D	T_w	T_c	T_h	μ	ρ	λ	c_p	
4	2	0.2	0.15	3	1	0.05	150	20	300	0.01	1	0.5	1000	Gas
										0.1	100	1	4000	Pipe

Table 2: Geometric and numerical parameters used in the 3D serpentine heater setup. All values in SI units, with the exception of temperature given in Celsius.

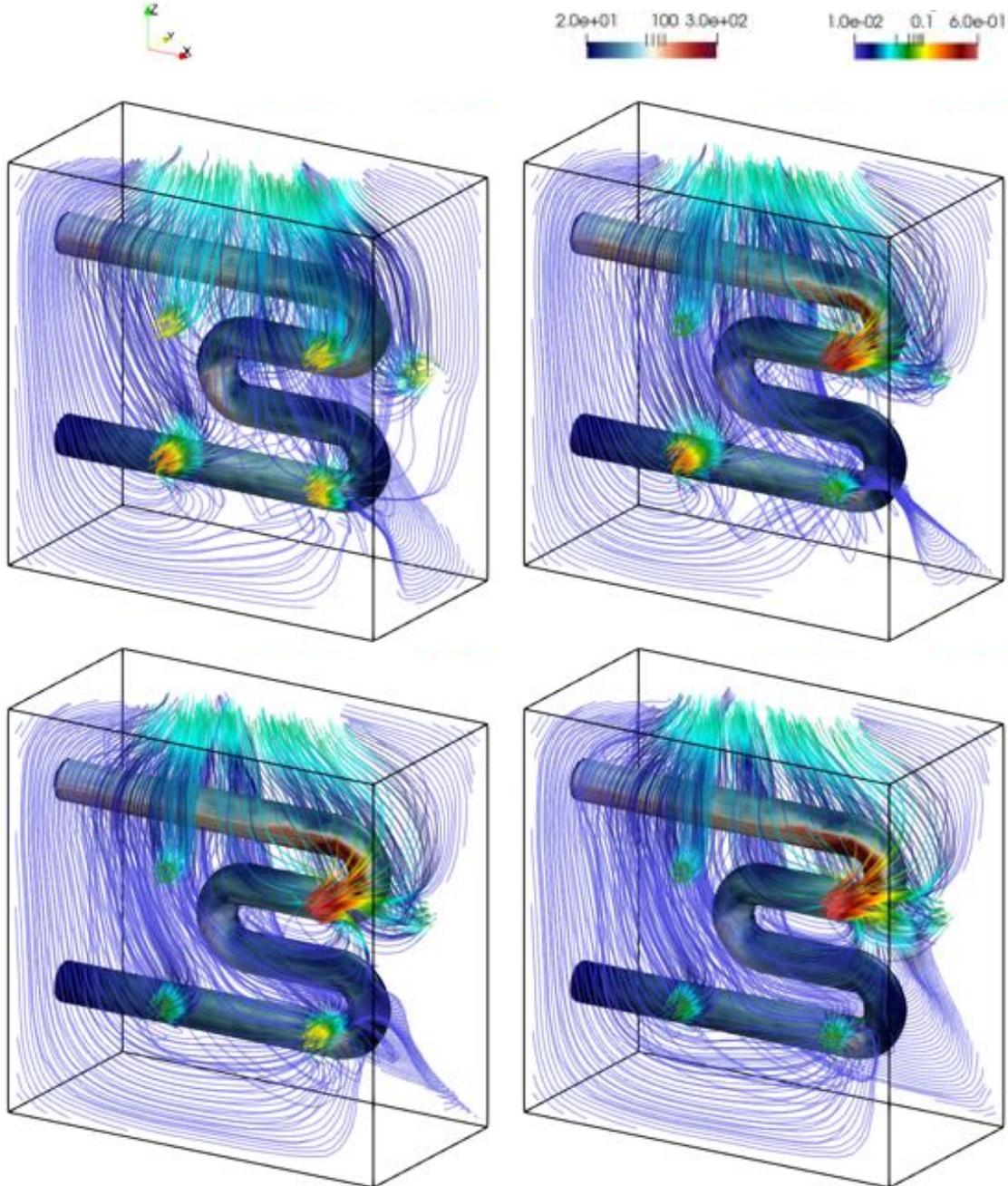


Figure 14: Representative temperature fields sampled on the course of optimization on the surface of the pipe together with streamlines colored by the magnitude of velocity, both in logarithmic scales.

397 5.2. Control strategy

398 The quantity being optimized is the distribution of the inflow between the 6 injectors $V_{i \in [1,6]}$.
399 In order to avoid asymmetry in the learning process we decided to control each of the injectors in

400 the same way, that is to assign each injector a value between 0.1 and 0.9, scaled a posteriori to
 401 recover a valid velocity satisfying (21). In order to compute the reward r_t we distribute 29 probes
 402 uniformly on the outlet of the pipe each at a distance $\Delta = 0.06$ of their closest neighbour, see
 403 figure 12 (b). These probes allow us to monitor the temperature out of the pipe, and the reward
 404 given the PPO algorithm (computed at the final simulation time) is defined as

$$r_t = \frac{1}{29} \sum_{i=1}^{29} T_i, \quad (22)$$

405 with T_i being the temperature at probe i .

406 In a second stage, we add a strong constraint to this reward. By looking at the highest temper-
 407 ature in the pipe across all time steps, we penalize the solutions that reach a temperature above a
 408 limit $T_l = 200$. We chose the limit by looking at the solutions found by the PPO algorithm with
 409 the first reward and their highest temperature inside the pipe. The new reward is thus

$$\rho_t = \begin{cases} r_t, & \text{if } \max_{\Pi} T < T_l, \\ 0, & \text{if } \max_{\Pi} T \geq T_l, \end{cases} \quad (23)$$

410 where Π is the inner pipe domain. This is meant to avoid concentrating too much energy in the
 411 same spot and thus protecting the materials used in the heating process.

412 5.3. Results

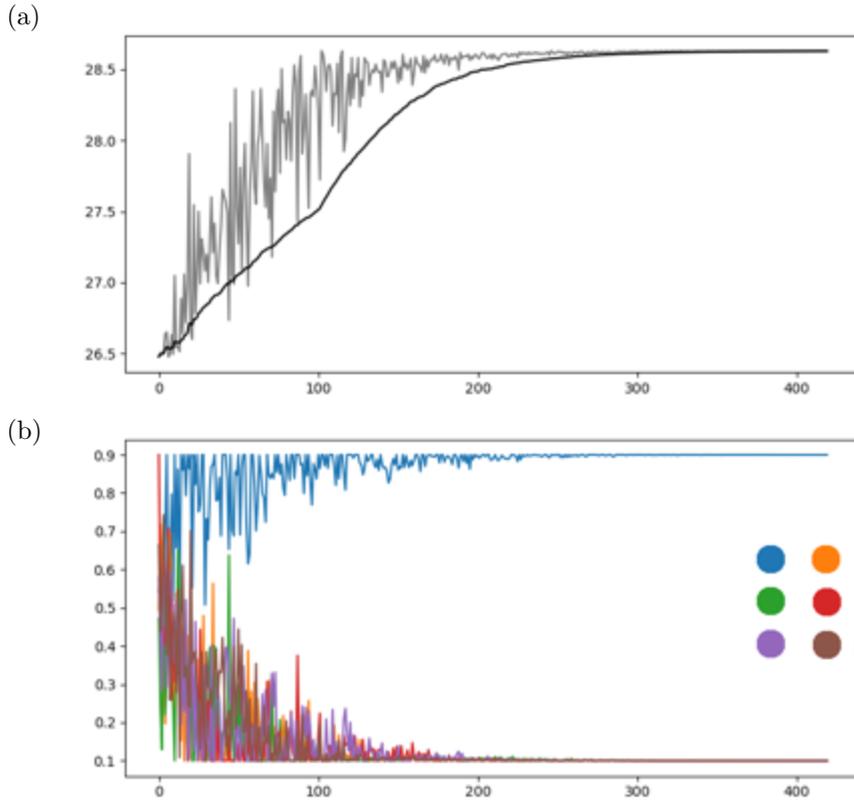


Figure 15: Evolution per episode of the (a) reward r_t , and (b) control velocity at each gas burner, and (c) maximum temperature $\max_{\Pi} T$ in the pipe. Black curves are the moving averages. The color code of the inlets is the same as in fig.12

413 For this case, 150 episodes have been run, each composed of 8 environment, each of which
 414 performs 1300 iterations with time step $\Delta t = 0.1$ to march in time the same initial condition
 415 (consisting of zero velocity and uniform temperature, except in the pipe domain). This represents

416 1200 simulations, each of which is performed on 8 cores and lasts 40mn, hence 800h of total CPU
 417 cost; see figure 14 for representative temperature and velocity distributions sampled of the course
 418 of optimization.

419 Figure 15 shows the evolution of the reward r_t and the values given to each inlet at each
 420 individual. The colors correspond to each inlet and are kept also in figure 16. The trend of these
 421 curves demonstrate a fast convergence of PBO in this context, with a transitional phase during
 422 which it tries to explore other possibilities. The solution found by the algorithm puts all the
 423 energy in a single, well positioned hot gas inlet (which is rather intuitive except that the location
 424 of said inlet may not be), as the last 200 episodes all yield the same distribution among the inlets : all
 425 for the top left inlet and nothing for the rest. As can be seen in figure 12 (a), this particular inlet is
 426 near the end of the tube, right in front of large area thanks to the bend in the tube. This position
 427 allows for a large part of the heat to be transferred to the tube without too much energy being lost
 428 in perturbations. The results of the learning process under the constrained reward ρ_t are shown
 429 in figure 16, where the red line marks the limit temperature T_l that we try not to top (which the
 430 single-step PPO algorithm successfully achieves, as the temperature converges to a value slightly
 431 below). No moving average is given here since the hard constraint forces the reward to be zero
 432 when the limit temperature is topped. A trend is however visible since the reward still goes up in
 433 general and the frequency of constraint violation reduces along the training. Colored curves can
 434 be linked to the figures 12 (a) and 15. After convergence, the DRL still puts most of the weight
 435 on the top left inlet, with the rest of energy mainly given to the top right inlet, right next to the
 436 other, and closer to the output of the pipe. This shows the ability of the method to find optimal
 437 solutions under the constraint of safe operating conditions (here the pipe maximum temperature).

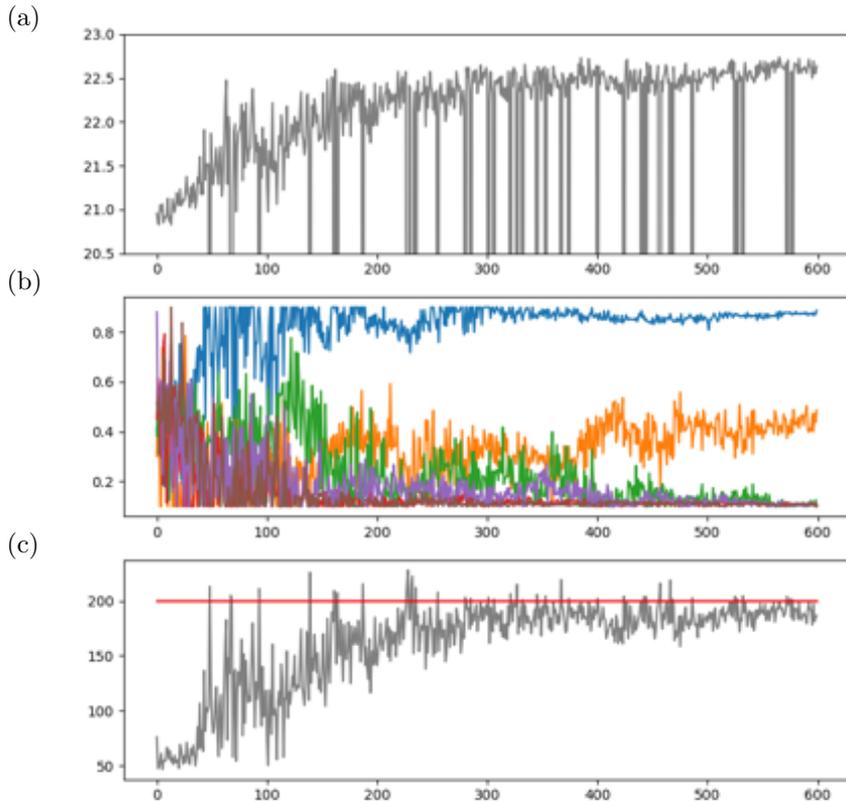


Figure 16: Evolution per episode of the (a) reward ρ_t , (b) control velocity at each gas burner, and (c) maximum temperature $\max_{\Pi} T$ in the pipe. Black curves are the moving averages. The color code of the inlets is the same as in fig. 12

438 5.4. Discussion

439 The main interest behind this comparison is to assess how the algorithm performs when a hard
440 constrained is applied directly to the reward. The first control experiment are somehow consistent
441 with intuition, but the second one provides with more relevant and exploitable results in a real
442 context where the ability to not exceed a certain temperature can be essential for the industrial
443 process. On another note, for this case, it could be more useful for this case to investigate the
444 feasibility to learn active control strategies (by adjusting dynamically the control velocities to
445 appropriate sensing of flow variables). As the temperature of one spot approaches the maximum
446 authorized value, this would allow redistributing some of the power to the other spots to balance
447 everything properly, and to converge rapidly to an optimal configuration. This would also allow
448 for some evolution of the input flow of the pipe, and a reaction to it.

449 6. General discussion

450 Several points deserve further consideration to keep pushing forward the development of DRL
451 in the context of such real-life applications. First, improving efficiency and convergence (by fine-
452 tuning the hyper parameters and comparing with different DRL algorithms allowing for increased
453 exploration, as PPO prevents by design large updates of the policy to avoid performance collapse).
454 Second, designing improved reward construction strategies, as approximating the reward from
455 point-wise temperature data has been shown to yield a certain sensitivity to system and numerical
456 uncertainty, which in turn may trap the algorithm in local optimal. Third, enriching the description
457 of the test cases using multi-physics modeling, *e.g.*, radiative heat transfer and thermo-mechanical
458 coupling to encompass the solid deformations. Finally, investigating the case of active flow control
459 in which the control parameters are dynamically adjusted from measurements in the workpiece
460 and the furnace, and has often proved to be of great importance in industrial contexts.

461 7. Conclusion

462 Optimization of heating processes is achieved here training a fully connected network with
463 the PBO deep reinforcement algorithm, in which it gets only one attempt per learning episode
464 at finding the optimal. The numerical reward fed to the network is computed with a stabilized
465 finite elements CFD environment solving the coupled Navier-Stokes and heat equations, using a
466 combination of variational multi-scale modeling, immerse volume method, and multi-component
467 anisotropic mesh adaptation.

468 The approach succeeds at improving the homogeneity of temperature (or a blend of homo-
469 geneity and absolute temperature) across the surface of two-dimensional cold workpieces under
470 jet impingement heating. Several control scenarios have been considered (that can be considered
471 different levels of design constraint), from the simple case where only the inflow velocity of the
472 hot air injectors is optimized relative to a fixed workpiece position, up to the most complex case
473 where the DRL agent also optimizes the position and insertion angle of the workpiece itself. The
474 potential of the approach for industrial configurations of engineering interest is also showcased by
475 optimizing the inflow of multiple gas burners in a three-dimensional serpentine heater.

476 The present results highlight the capabilities of coupling DRL and computational fluid dynam-
477 ics in the context of industrial manufacturing processes in general, and heating processes inside
478 industrial furnaces in particular.

479 References

- 480 [1] S. Dequan, G. Guili, G. Zhiwei, X. Peng, [Application of Expert Fuzzy PID Method for](#)
481 [Temperature Control of Heating Furnace](#), *Procedia Engineering* 29 (2012) 257–261. doi:
482 [10.1016/j.proeng.2011.12.703](#),
483 URL <https://www.sciencedirect.com/science/article/pii/S1877705811065404>

- 484 [2] M. Tóthová, M. Balara, J. Dubják, [Simulation Model of Cascade Control of the Heating](#)
485 [System](#), International Journal of Engineering Research in Africa 18 (2015) 20–27, conference
486 Name: International Journal of Engineering Research in Africa Vol. 18 ISBN: 9783038356790
487 Publisher: Trans Tech Publications Ltd. [doi:10.4028/www.scientific.net/JERA.18.20](#),
488 URL <https://www.scientific.net/JERA.18.20>
- 489 [3] N. Philip, S. Sahlan, A. P. I. D. N. Wahab, Application of Auto-Tuner Fuzzy PID Controller
490 on Industrial Cascade Control, ELEKTRIKA- Journal of Electrical Engineering 19 (2020)
491 61–65.
- 492 [4] E. Rafajłowicz, W. Rafajłowicz, Image-Driven Decision Making with Application to Control
493 Gas Burners, in: K. Saeed, W. Homenda, R. Chaki (Eds.), Computer Information Systems
494 and Industrial Management, Lecture Notes in Computer Science, Springer International Pub-
495 lishing, Cham, 2017, pp. 436–446. [doi:10.1007/978-3-319-59105-6_37](#).
- 496 [5] E. Rafajłowicz, H. Pawlak-Kruczek, W. Rafajłowicz, Statistical Classifier with Ordered De-
497 cisions as an Image Based Controller with Application to Gas Burners, in: L. Rutkowski,
498 M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, J. M. Zurada (Eds.), Artificial
499 Intelligence and Soft Computing, Lecture Notes in Computer Science, Springer International
500 Publishing, Cham, 2014, pp. 586–597. [doi:10.1007/978-3-319-07173-2_50](#).
- 501 [6] N. Nguyen, D. Tran, Neural Network based Model Reference Control for Electric Heating
502 Furnace with Input Saturation, 2019, pp. 111–114. [doi:10.1109/ICA-SYMP.2019.8646052](#).
- 503 [7] D.-D. Wen, Decoupling control of electric heating furnace temperature based on DRNN neural
504 network, 2010, pp. 261–264. [doi:10.1109/ICECTECH.2010.5479934](#).
- 505 [8] V. R. Radhakrishnan, A. R. Mohamed, [Neural networks for the identification and control](#)
506 [of blast furnace hot metal quality](#), Journal of Process Control 10 (6) (2000) 509–524. [doi:](#)
507 [10.1016/S0959-1524\(99\)00052-9](#),
508 URL <https://www.sciencedirect.com/science/article/pii/S0959152499000529>
- 509 [9] L. Zhang, Y. Xue, Q. Xie, Z. Ren, [Analysis and neural network prediction of combustion](#)
510 [stability for industrial gases](#), Fuel 287 (2021) 119507. [doi:10.1016/j.fuel.2020.119507](#),
511 URL <https://www.sciencedirect.com/science/article/pii/S0016236120325035>
- 512 [10] J. Matthes, P. Waibel, M. Vogelbacher, H. J. Gehrmann, H. B. Keller, [A new camera-based](#)
513 [method for measuring the flame stability of non-oscillating and oscillating combustions](#), Ex-
514 perimental Thermal and Fluid Science 105 (2019) 27–34. [doi:10.1016/j.expthermflusci.](#)
515 [2019.03.008](#),
516 URL <https://www.sciencedirect.com/science/article/pii/S0894177718317503>
- 517 [11] J. Matthes, P. Waibel, M. Vogelbacher, H. B. Keller, H. J. Gehrmann, D. Stapf, A camera-
518 based flame stability controller for non-oscillating and forced-oscillating combustion, in: Pro-
519 ceedings of the 13TH EUROPEAN CONFERENCE ON INDUSTRIAL FURNACES AND
520 BOILERS, 2022.
- 521 [12] J. Tao, Z. Yu, R. Zhang, F. Gao, [RBF neural network modeling approach using PCA](#)
522 [based LM-GA optimization for coke furnace system](#), Applied Soft Computing 111 (2021)
523 107691. [doi:10.1016/j.asoc.2021.107691](#),
524 URL <https://www.sciencedirect.com/science/article/pii/S1568494621006128>
- 525 [13] D. Zhao, Z. Lu, H. Zhao, X. Y. Li, B. Wang, P. Liu, [A review of active control approaches](#)
526 [in stabilizing combustion systems in aerospace industry](#), Progress in Aerospace Sciences 97
527 (2018) 35–60. [doi:10.1016/j.paerosci.2018.01.002](#),
528 URL <https://www.sciencedirect.com/science/article/pii/S0376042117300878>
- 529 [14] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert,
530 L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driess-
531 che, T. Graepel, D. Hassabis, [Mastering the game of Go without human knowledge](#), Na-
532 ture 550 (7676) (2017) 354–359, number: 7676 Publisher: Nature Publishing Group. [doi:](#)

-
- 533 [10.1038/nature24270](https://doi.org/10.1038/nature24270).
534 URL <https://www.nature.com/articles/nature24270>
- 535 [15] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller,
536 [Playing Atari with Deep Reinforcement Learning](https://arxiv.org/abs/1312.5602), arXiv:1312.5602 [cs]ArXiv: 1312.5602 (Dec.
537 2013).
538 URL <http://arxiv.org/abs/1312.5602>
- 539 [16] [Google just gave control over data center cooling to an AI](https://www.technologyreview.com/2018/08/17/140987/google-just-gave-control-over-data-center-cooling-to-an-ai/).
540 URL [https://www.technologyreview.com/2018/08/17/140987/
541 google-just-gave-control-over-data-center-cooling-to-an-ai/](https://www.technologyreview.com/2018/08/17/140987/google-just-gave-control-over-data-center-cooling-to-an-ai/)
- 542 [17] A. Gupta, A. Khwaja, L. Guan, B. Venkatesh, Policy-Gradient and Actor-Critic Based State
543 Representation Learning for Safe Driving of Autonomous Vehicles, Sensors 20 (2020) 5991.
544 [doi:10.3390/s20215991](https://doi.org/10.3390/s20215991).
- 545 [18] F. Wang, L. P. Casalino, D. Khullar, Deep Learning in Medicine-Promise, Progress, and
546 Challenges, JAMA internal medicine 179 (3) (2019) 293–294. [doi:10.1001/jamainternmed.
547 2018.7117](https://doi.org/10.1001/jamainternmed.2018.7117).
- 548 [19] D. Skrobek, J. Krzywanski, M. Sosnowski, A. Kulakowska, A. Zylka, K. Grabowska, K. Ciesiel-
549 ska, W. Nowak, [Prediction of sorption processes using the deep learning methods \(long short-
550 term memory\)](https://www.mdpi.com/1996-1073/13/24/6601), Energies 13 (24) (2020). [doi:10.3390/en13246601](https://doi.org/10.3390/en13246601).
551 URL <https://www.mdpi.com/1996-1073/13/24/6601>
- 552 [20] J. Krzywanski, K. Sztokler, M. Szubel, T. Siwek, W. Nowak, Mika, [A comprehensive three-
553 dimensional analysis of a large-scale multi-fuel cfb boiler burning coal and syngas. part 1. the
554 cfd model of a large-scale multi-fuel cfb combustion](https://www.mdpi.com/1099-4300/22/9/964), Entropy 22 (9) (2020). [doi:10.3390/
555 e22090964](https://doi.org/10.3390/e22090964).
556 URL <https://www.mdpi.com/1099-4300/22/9/964>
- 557 [21] P. Garnier, J. Viquerat, J. Rabault, A. Larcher, A. Kuhnle, E. Hachem, [A review on deep
558 reinforcement learning for fluid mechanics](https://www.sciencedirect.com/science/article/pii/S0045793021001407), Computers & Fluids 225 (2021) 104973. [doi:
559 10.1016/j.compfluid.2021.104973](https://doi.org/10.1016/j.compfluid.2021.104973).
560 URL <https://www.sciencedirect.com/science/article/pii/S0045793021001407>
- 561 [22] J. Viquerat, P. Meliga, A. Larcher, E. Hachem, A review on deep reinforcement learning for
562 fluid mechanics : an update, Phys. Fluids 34 (2022) 111301.
- 563 [23] G. Beintema, A. Corbetta, L. Biferale, F. Toschi, Controlling Rayleigh-B\'enard convection
564 via reinforcement learning, arXiv preprint arXiv:2003.14358 (2020).
- 565 [24] E. Hachem, H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga, [Deep reinforcement learning for
566 the control of conjugate heat transfer](https://www.sciencedirect.com/science/article/pii/S0021999121002126), Journal of Computational Physics 436 (2021) 110317.
567 [doi:10.1016/j.jcp.2021.110317](https://doi.org/10.1016/j.jcp.2021.110317).
568 URL <https://www.sciencedirect.com/science/article/pii/S0021999121002126>
- 569 [25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, [Proximal Policy Optimization
570 Algorithms](https://arxiv.org/abs/1707.06347), arXiv:1707.06347 [cs]ArXiv: 1707.06347 (Aug. 2017).
571 URL <http://arxiv.org/abs/1707.06347>
- 572 [26] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, A. Larcher, E. Hachem, [Direct shape opti-
573 mization through deep reinforcement learning](https://arxiv.org/abs/1908.09885), arXiv:1908.09885 [cs]ArXiv: 1908.09885 (Dec.
574 2020).
575 URL <http://arxiv.org/abs/1908.09885>
- 576 [27] H. Ghraieb, J. Viquerat, A. Larcher, P. Meliga, E. Hachem, Single-step deep reinforcement
577 learning for open-loop control of laminar and turbulent flows, Phys. Rev. Fluids 6 (2021)
578 053902.

- 579 [28] E. Hachem, H. Dignonnet, E. Massoni, T. Coupez, Immersed volume method for solving natural
580 convection, conduction and radiation of a hat-shaped disk inside a 3d enclosure, International
581 Journal of numerical methods for heat & fluid flow (2012).
- 582 [29] E. Hachem, B. Rivaux, T. Kloczko, H. Dignonnet, T. Coupez, [Stabilized finite element method](#)
583 [for incompressible flows with high reynolds number](#), Journal of Computational Physics 229 (23)
584 (2010) 8643–8665. [doi:https://doi.org/10.1016/j.jcp.2010.07.030](https://doi.org/10.1016/j.jcp.2010.07.030),
585 URL <https://www.sciencedirect.com/science/article/pii/S0021999110004237>
- 586 [30] E. Hachem, T. Kloczko, H. Dignonnet, T. Coupez, [Stabilized finite element solution to han-](#)
587 [dle complex heat and fluid flows in industrial furnaces using the immersed volume method](#),
588 International Journal for Numerical Methods in Fluids 68 (1) (2012) 99–121. [arXiv:](#)
589 <https://onlinelibrary.wiley.com/doi/pdf/10.1002/flid.2498>, [doi:https://doi.org/](https://doi.org/10.1002/flid.2498)
590 [10.1002/flid.2498](https://doi.org/10.1002/flid.2498),
591 URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/flid.2498>
- 592 [31] I. Goodfellow, Y. Bengio, A. Courville, The Deep Learning Book, MIT Press, 2017.
- 593 [32] R. S. Sutton, A. G. Barto, Reinforcement learning: An introduction, MIT press, 2018.
- 594 [33] D. Rumelhart, G. Hinton, R. Williams, Learning representations by back-propagating errors,
595 Nature 323 (1986) 533–536.
- 596 [34] A. Kakade, A natural policy gradient, Adv. Neural Inf. Process Syst. 14 (2001) 1531–1538.
- 597 [35] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, P. Abbeel, Trust Region Policy Optimization,
598 arXiv e-prints (Feb. 2015). [arXiv:1502.05477](#)
- 599 [36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal Policy Optimization
600 Algorithms, arXiv e-prints (Jul. 2017). [arXiv:1707.06347](#)
- 601 [37] T. Coupez, L. Silva, E. Hachem, [Implicit Boundary and Adaptive Anisotropic](#)
602 [Meshing](#), Springer International Publishing, Cham, 2015, pp. 1–18. [doi:10.1007/](https://doi.org/10.1007/978-3-319-06053-8_1)
603 [978-3-319-06053-8_1](https://doi.org/10.1007/978-3-319-06053-8_1),
604 URL https://doi.org/10.1007/978-3-319-06053-8_1
- 605 [38] T. W. Athan, P. Y. Papalambros, A note on weighted criteria methods for compromise solu-
606 tions in multi-objective optimization, Eng. Optim. 27 (1996) 155–176.
- 607 [39] J. Viquerat, R. Duvigneau, P. Meliga, A. Kuhnle, E. Hachem, Policy-based optimization:
608 single-step policy gradient method seen as an evolution strategy, Neural Comput. Appl. (2022).