



HAL
open science

Deep reinforcement learning for the control of conjugate heat transfer

Elie Hachem, Hassan Ghraieb, Jonathan Viquerat, Aurélien Larcher, P. Meliga

► **To cite this version:**

Elie Hachem, Hassan Ghraieb, Jonathan Viquerat, Aurélien Larcher, P. Meliga. Deep reinforcement learning for the control of conjugate heat transfer. 2020. hal-03027923v1

HAL Id: hal-03027923

<https://hal.science/hal-03027923v1>

Preprint submitted on 27 Nov 2020 (v1), last revised 17 Nov 2021 (v2)

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.

Deep reinforcement learning for the control of conjugate heat transfer

E. Hachem^{a,*}, H. Ghraieb^a, J. Viquerat^a, A. Larcher^a, P. Meliga^a

^a*MINES ParisTech, PSL Research University, Centre de mise en forme des matériaux (CEMEF), CNRS UMR 7635, 06904 Sophia Antipolis Cedex, France*

Abstract

This research gauges the ability of deep reinforcement learning (DRL) techniques to assist the control of conjugate heat transfer systems. It uses a novel, “degenerate” version of the proximal policy optimization (PPO) algorithm to train a neural network in optimizing said system only once per learning episode, and an in-house stabilized finite elements environment combining variational multiscale (VMS) modeling of the governing equations, immerse volume method, and multi-component anisotropic mesh adaptation to compute the numerical reward fed to the neural network. Several test cases of natural and forced convection are used as testbed for developing the methodology, that proves successful to alleviate the heat transfer enhancement related to the onset of convection in a two-dimensional, differentially heated square cavity, and to improve the homogeneity of temperature across the surface of two and three-dimensional hot workpieces under impingement cooling. Beyond adding value to the shallow literature on this subject, these findings establish the potential of single-step PPO for reliable black-box optimization of computational fluid dynamics (CFD) systems, and pave the way for future progress in the optimal control of conjugate heat transfer using this new class of methods.

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

1. Introduction

Thermal control, defined as the ability to finesse the thermal properties of a volume of fluid (and of the solid objects inside) into a certain desired state, is a field of tremendous societal and economical importance. For instance, heat/cool exchangers are used in a broad range of industrial applications to regulate process temperatures by heat or cool transfer between fluid media, which in turn ensures that machinery, chemicals, water, gas, and other substances remain within safe operating conditions. Green building engineering is another field relying on such ability to manage indoor thermal conditions (temperature, humidity) under substantial variations of the ambient conditions to provide high-quality living and working environments. In many manufacturing processes, thermal conditioning is also intended to improve the final mechanical (e.g., hardness, toughness, resistance), electrical, or optical properties of the product, the general picture being that high temperature gradients are useful to speed up the process but generally harm the quality of the outcome because of heat transfer inhomogeneities caused by the increased convection by the fluid particles. All such problems fall under the purview of this line of study.

Numerous strategies have been implemented to control fluid mechanical systems (including conjugate heat transfer systems combining thermal conduction in the solid and convective transfer in the fluid), either open-loop with passive appendices (e.g., end plate, splitter plate, small secondary cylinder, or flexible tail), or open-loop with actuating devices (e.g., plasma actuation, boundary temperatures, steady or unsteady base bleeding, rotation) or closed-loop (e.g. via transverse motion, perturbations of the thermal boundary layer, blowing/suction, rotation, all relying on an appropriate sensing of flow variables). Nonetheless, many of the proposed strategies are trial and

*Corresponding author

Email address: `elie.hachem@mines-paristech.fr` (E. Hachem)

22 error, and therefore require extensive and costly experimental or numerical campaigns. This has
23 motivated the development of analytical methods and numerical algorithms for the optimal control
24 of Navier–Stokes systems [1–3], and the maturing of mathematical methods in flow control and
25 discrete concepts for PDE constrained optimization. Applications to the heat equation [4] and the
26 coupled Navier–Stokes and heat equations [5–8] have also been considered, including fresh devel-
27 opments meant to alter the linear amplification of flow disturbances [9], but the general picture
28 remains that the optimal control of conducting, convecting fluids has not been extensively studied.

29 The premise of this research is that the related task of selecting an optimal subset of control
30 parameters can alternatively be assisted by machine learning algorithms. Indeed, the introduction
31 of the back-propagation algorithm [10] has progressively turned Artificial Neural Networks (ANN)
32 into a family of versatile non-parametric tools that can be trained to hierarchically extract informa-
33 tive features from data and to provide qualitative and quantitative modeling predictions. Together
34 with the increased affordability of high performance computational hardware, this has allowed
35 leveraging the ever-increasing volume of data generated for research and engineering purposes into
36 novel insight and actionable information, which in turn has entirely transformed scientific disci-
37 plines, such as robotics [11, 12] or image analysis [13]. Owing to the ability of neural networks
38 to handle stiff, large-scale nonlinear problems [14], machine learning algorithms have also been
39 making rapid inroads in fluid mechanics, as a mean to solve the Navier–Stokes equations [15] or
40 to predict closure terms in turbulence models [16]; see also Ref. [17] for an overview of the current
41 developments and opportunities.

42 Neural networks can also be used to solve decision-making problems, which is the purpose of
43 Deep Reinforcement Learning (DRL, where the *deep* terminology generally weighs on the sizable
44 depth of the network), an advanced branch of machine learning. Simply put, a neural network
45 trains in finding out which actions or succession of actions maximize a numerical reward signal,
46 with the possibility for a given action to affect not only the immediate but also the future rewards.
47 Successful applications of DRL range from AlphaGo, the well-known ANN that defeated the top-
48 level human player at the game of Go [18] to the real-world deployment of legged robots [19], to
49 breakthroughs in computer vision (e.g., filtering, or extracting image features) [20] and optimal
50 control problems [21, 22]. Despite the many achievements, DRL remains surprisingly sparsely
51 used in fluid mechanics, with a limited amount of literature barely scratching the surface of the
52 performance improvements to be delivered in low-Reynolds-number flow control [23–25] and shape
53 optimization [26]. The literature on thermal control is even more sparse, with only Ref. [27]
54 addressing the control of natural convection dominated heat transfer (this is a recent effort similar
55 to the present work, conducted in the same time frame, that we became aware of during the
56 redaction of this manuscript), plus a few other publications dealing with energy efficiency in civil
57 engineering from low-dimensional thermodynamic models basically unrelated to the equations of
58 fluid dynamics [28, 29].

59 This research assesses the feasibility of using proximal policy optimization (PPO [22]) for control
60 and optimization purposes of conjugate heat transfer systems, as governed by the coupled Navier–
61 Stokes and heat equations. The objective here is to keep shaping the capabilities of the method
62 (PPO is still a relatively newcomer that has quickly emerged as the go-to DRL algorithm due to
63 its data efficiency, simplicity of implementation and reliable performance) and to narrow the gap
64 between DRL and advanced numerical methods for multiscale, multi-physics computational fluid
65 dynamics (CFD). We investigate more specifically the “degenerate” single-step PPO algorithm
66 introduced in [26], in which the neural network gets only one attempt per learning episode at
67 finding the optimal. Several problems of conjugate heat transfer in two and three dimensions are
68 used as testbed to push forward the development of this novel approach, whose high potential as
69 a reliable black-box optimizer for CFD problems (where only the final configuration is of interest)
70 has been recently assessed for aerodynamic applications [30]. To the best of the authors knowledge,
71 this constitutes the first attempt to achieve DRL-based control of *forced* convection (with [27] being
72 the first attempt to achieve DRL control of *natural* convection, to give credit where it is due).

73 The organization is as follows: section 2 outlines the main features of the finite element CFD
74 environment used to compute the numerical reward fed to the neural network, that combines
75 variational multiscale (VMS) modeling of the governing equations, immerse volume method, and
76 multi-component anisotropic mesh adaptation. The baseline principles and assumptions of DRL
77 and PPO are presented in section 3, together with the specifics of the single-step PPO algorithm.

78 Section 4 revisits the natural convection case of [27] for the purpose of validation and assessment
79 part of the method capabilities. In section 5, DRL is used to control conjugate heat transfer in
80 a model setup of two-dimensional workpiece cooling by impingement of a fluid. An extension to
81 three-dimensional workpieces is proposed in section 6.

82 2. Computational fluid dynamics

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

$$\nabla \cdot \mathbf{u} = 0, \quad (1)$$

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

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

86 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
87 rate of deformation tensor, $\boldsymbol{\psi}$ and χ are source terms (modeling, e.g., buoyancy or radiative heat
88 transfer), and we assume here constant fluid density ρ , dynamic viscosity μ , thermal conductivity
89 λ , and specific heat c_p .

90 2.1. The immersed volume method

91 The numerical modeling of conjugate heat transfer mostly depends upon a heat transfer co-
92 efficient to ensure that the proper amount of heat is exchanged at the fluid/solid interface via
93 thermal boundary conditions. Computing said coefficient is no small task (as it requires solving
94 an inverse problem to assimilate experimental data, which in turn requires relevant experimental
95 data to be available), and is generally acknowledged to be a limiting issue for practical applica-
96 tions where one must vary, e.g., the shape, number and position of the solid, or the fluid and/or
97 solid material properties. We thus rather use here the immersed volume method (IVM) to combine
98 both the fluid and solid phases into a single fluid with variable material properties. Simply put,
99 we solve equations formally identical to (1)-(3) on a unique computational domain Ω , but with
100 variable density, dynamic viscosity, conductivity, and specific heat, which removes the need for a
101 heat transfer coefficient since the amount of heat exchanged at the interface then proceeds solely
102 from the individual material properties on either side of it. In order to ensure numerical accu-
103 racy, such an approach must combine three key ingredients, that are briefly reviewed in the next
104 paragraphs: an interface capturing method, anisotropic mesh adaptation to achieve a high-fidelity
105 description of said interface, and relevant mixing laws to describe the properties of the composite
106 fluid. One point worth being mentioned is that the interface here is static, although the same
107 numerical framework can be used to dynamically track moving interfaces, and thus to encompass
108 the effect of solid displacements. This is because the solid is fixed once an action has been taken
109 by the PPO agent, although not fixed over the course of optimization, as the solid position can
110 very well be the quantity subjected to optimization, as illustrated in section 5.3.4.

112 - *Level set method*:. the level set approach is used to localize the fluid/solid interface by the zero
113 iso-value of a smooth function. In practice, a signed distance function ϕ is used to localize the
114 interface and initialize the material properties on both either side of it, with the convention that
115 $\phi > 0$ (res. $\phi < 0$) in the fluid (resp. the solid).

117 - *Anisotropic mesh adaptation*:. the interface may intersect arbitrarily the mesh elements if it
118 is not aligned with the element edges, in which case discontinuous material properties across
119 the interface can yield oscillations of the numerical solutions. We thus use the anisotropic mesh
120 adaptation technique presented in [31] to ensure that the material properties are distributed as
121 accurately and smoothly as possible over the smallest possible thickness around the interface. This
122 is done computing modified distances from a symmetric positive defined tensor (the metric) whose
123 eigenvectors define preferential directions along which mesh sizes can be prescribed from the related

124 eigenvalues. The metric used here is isotropic far from the interface, with mesh size set equal to
 125 h_∞ in all directions, but anisotropic near the interface, with mesh size equal to h_\perp in the direction
 126 normal to the interface, and to h_∞ in the other directions, which can be written for an intended
 127 thickness δ as

$$128 \quad \mathbf{M} = K(\phi)\mathbf{n} \otimes \mathbf{n} + \frac{1}{h_\infty}\mathbf{I} \quad \text{with} \quad K(\phi) = \begin{cases} 0 & \text{if } |\phi| \geq \delta/2, \\ \frac{1}{h_\perp^2} - \frac{1}{h_\infty^2} & \text{if } |\phi| < \delta/2, \end{cases} \quad (4)$$

128 where $\mathbf{n} = \nabla\phi/|\nabla\phi|$ is the unit normal to the fluid/solid interface computed from the level set
 129 gradient. A posteriori anisotropic error estimator is then used to minimize the interpolation error
 130 under the constraint of a fixed number of edges in the mesh. A unique metric can be built from
 131 multi-component error vectors [31–34], which is especially relevant for conjugate heat transfer op-
 132 timization, as it allows each learning episode to use an equally accurate mesh adapted from the
 133 velocity vector and magnitude, the temperature field, and the level set.

134
 135 - *Mixing laws*:. the composite density, dynamic viscosity and specific heat featured in equations (1)-
 136 (3) are computed by linear interpolation of the fluid and solid values, for instance the composite
 137 density is

$$\rho = \rho_f H_\epsilon(\phi) + \rho_s(1 - H_\epsilon(\phi)), \quad (5)$$

138 where H_ϵ is the smoothed Heaviside function defined as

$$H_\epsilon(\phi) = \begin{cases} 0 & \text{if } \phi < -\epsilon, \\ \frac{1}{2}\left(1 + \frac{\phi}{\epsilon} + \frac{1}{\pi}\sin\left(\pi\frac{\phi}{\epsilon}\right)\right) & \text{if } |\phi| \leq \epsilon, \\ 1 & \text{if } \phi > \epsilon, \end{cases} \quad (6)$$

139 and ϵ is a regularization parameter proportional to the mesh size in the normal direction to the
 140 interface, set here to $\epsilon = 2h_\perp$. Doing so for the thermal conductivity would however lead to inac-
 141 curate results [35], hence the inverse of the thermal conductivity is linearly interpolated according
 142 to

$$\frac{1}{\lambda} = \frac{1}{\lambda_f} H_\epsilon(\phi) + \frac{1}{\lambda_s} (1 - H_\epsilon(\phi)), \quad (7)$$

143 to ensure conservation of the heat flux.

144 2.2. Variational multiscale approach (VMS)

145 In the context of finite element methods (that remain widely used to simulate engineering
 146 CFD systems due to their ability to handle complex geometries), direct numerical simulation
 147 (DNS) solves the weak form of (1)-(3), obtained by integrating by parts the pressure, viscous and
 148 conductive terms, to give

$$(\rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}), \mathbf{w}) + (2\mu \boldsymbol{\varepsilon}(\mathbf{u}), \boldsymbol{\varepsilon}(\mathbf{w})) - (p, \nabla \cdot \mathbf{w}) + (\nabla \cdot \mathbf{u}, q) = (\boldsymbol{\psi}, \mathbf{w}), \quad (8)$$

$$(\rho c_p (\partial_t T + \mathbf{u} \cdot \nabla T), s) + (\lambda \nabla T, \nabla s) = (\chi, s), \quad (9)$$

149 where (\cdot, \cdot) is the L^2 inner product on the computational domain, \mathbf{w} , q and s are relevant test
 150 functions for the velocity, pressure and temperature variables, and all fluid properties are those
 151 mixed with the smoothed Heaviside function defined in (6).

152 We use here the variational multiscale (VMS) approach [36–38] to solve a stabilized formulation
 153 of (8)-(9), which allows circumventing the Babuska–Brezzi condition (that otherwise imposes that
 154 different interpolation orders be used to discretize the velocity and pressure variables, while we
 155 use here simple continuous piecewise linear P_1 elements for all variables) and prevents numerical
 156 instabilities in convection regimes at high Reynolds numbers. We shall not go into the extensive
 157 details about the derivation of the stabilized formulations, for which the reader is referred to [39, 40].
 158 Suffice it to say here that the flow quantities are split into coarse and fine scale components, that

159 correspond to different levels of resolution. The fine scales are solved in an approximate manner
 160 to allow modeling their effect into the large-scale equations, which gives rise to additional terms in
 161 the right-hand side of (8)-(9), and yields the following weak forms for the large scale

$$\begin{aligned} & (\rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}), \mathbf{w}) + (2\mu \boldsymbol{\varepsilon}(\mathbf{u}), \boldsymbol{\varepsilon}(\mathbf{w})) - (p, \nabla \cdot \mathbf{w}) + (\nabla \cdot \mathbf{u}, q) = (\boldsymbol{\psi}, \mathbf{w}) \\ & \quad + \sum_{K \in \mathcal{T}_h} [(\tau_1 \mathcal{R}_M, \mathbf{u} \cdot \nabla \mathbf{w})_K + (\tau_1 \mathcal{R}_M, \nabla q)_K + (\tau_2 \mathcal{R}_C, \nabla \cdot \mathbf{w})_K], \end{aligned} \quad (10)$$

$$\begin{aligned} & (\rho c_p (\partial_t T + \mathbf{u} \cdot \nabla T), s) + (\lambda \nabla T, \nabla s) = (\chi, s) \\ & \quad + \sum_{K \in \mathcal{T}_h} [(\tau_3 \mathcal{R}_T, \mathbf{u} \cdot \nabla s)_K + (\tau_4 \mathcal{R}_T, \mathbf{u}_{\parallel} \cdot \nabla s)_K], \end{aligned} \quad (11)$$

162 where $(\cdot, \cdot)_K$ is the inner product on element K , and the \mathcal{R} terms are residuals for the governing
 163 equations defined by

$$-\mathcal{R}_C = \nabla \cdot \mathbf{u}, \quad -\mathcal{R}_M = \rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}) + \nabla p - \boldsymbol{\psi} \quad -\mathcal{R}_T = \rho c_p (\partial_t T + \mathbf{u} \cdot \nabla T) - \chi, \quad (12)$$

164 In (10), $\tau_{1,2}$ are ad-hoc mesh-dependent stabilization parameters defined in [41, 42]. In (11), \mathbf{u}_{\parallel} is
 165 the projection of the velocity along the direction of the temperature gradient, and $\tau_{3,4}$ are mesh-
 166 independent stabilization parameters acting both in the direction of the solution and of its gradient,
 167 that proceed from the stabilization of the ubiquitous convection-diffusion-reaction equation [43, 44],
 168 whose definition is given in [45, 46].

169 The governing equations are solved sequentially, i.e., we solve first (10), then use the resulting
 170 fluid velocity to solve (11). All linear systems are preconditioned with a block Jacobi method
 171 supplemented by an incomplete LU factorization, and solved with the GMRES algorithm, with
 172 tolerance threshold set to 10^{-6} for the Navier–Stokes equations, and 10^{-5} for the heat equation.
 173 The time derivatives, source and VMS stabilization terms are approximated explicitly with the
 174 forward Euler scheme. The viscous, pressure and divergence terms of the Navier–Stokes equations
 175 are integrated implicitly with the backward Euler scheme. The convection term is discretized
 176 semi-implicitly using the first-order backward Newton–Gregory formula, which yields

$$\begin{aligned} & (\rho(\frac{\mathbf{u}^{i+1} - \mathbf{u}^i}{\Delta t} + \mathbf{u}^i \cdot \nabla \mathbf{u}^{i+1}), \mathbf{w}) + (2\mu \boldsymbol{\varepsilon}(\mathbf{u}^{i+1}), \boldsymbol{\varepsilon}(\mathbf{w})) - (p^{i+1}, \nabla \cdot \mathbf{w}) + (\nabla \cdot \mathbf{u}^{i+1}, q) = (\boldsymbol{\psi}^i, \mathbf{w}) \\ & \quad + \sum_{K \in \mathcal{T}_h} [(\tau_1^i \mathcal{R}_M^{i+1}, \mathbf{u}^i \cdot \nabla \mathbf{w})_K + (\tau_1^i \mathcal{R}_M^{i+1}, \nabla q)_K + (\tau_2^i \mathcal{R}_C^{i+1}, \nabla \cdot \mathbf{w})_K], \end{aligned} \quad (13)$$

177 with residuals

$$-\mathcal{R}_C^{i+1} = \nabla \cdot \mathbf{u}^{i+1}, \quad -\mathcal{R}_M^{i+1} = \rho(\frac{\mathbf{u}^{i+1} - \mathbf{u}^i}{\Delta t} + \mathbf{u}^i \cdot \nabla \mathbf{u}^{i+1}) + \nabla p^{i+1} - \boldsymbol{\psi}^i, \quad (14)$$

178 where the i superscript refers to the solution at time $t_i = i\Delta t$. The convection and conduction
 179 terms of the heat equation are integrated implicitly with the backward Euler scheme, to give

$$\begin{aligned} & (\rho c_p (\frac{T^{i+1} - T^i}{\Delta t} + \mathbf{u}^{i+1} \cdot \nabla T^{i+1}), s) + (\lambda \nabla T^{i+1}, \nabla s) = (\chi^i, s) \\ & \quad + \sum_{K \in \mathcal{T}_h} [(\tau_3^i \mathcal{R}_T^{i+1}, \mathbf{u}^{i+1} \cdot \nabla s)_K + (\tau_4^i \mathcal{R}_T^{i+1}, \mathbf{u}_{\parallel}^{i+1} \cdot \nabla s)_K], \end{aligned} \quad (15)$$

180 with residual

$$-\mathcal{R}_T^{i+1} = \rho c_p (\frac{T^{i+1} - T^i}{\Delta t} + \mathbf{u}^{i+1} \cdot \nabla T^{i+1}) - \chi^i. \quad (16)$$

181 We solve (13)-(15) with an in-house VMS solver whose accuracy and reliability with respect to the
 182 intended application has been assessed in a series of previous papers; see especially [42, 47] for a
 183 detailed presentation of the mathematical formulation relevant to the IVM of a rigid body in an
 184 incompressible fluid and [40, 48] for an application to conjugate heat transfer analysis.

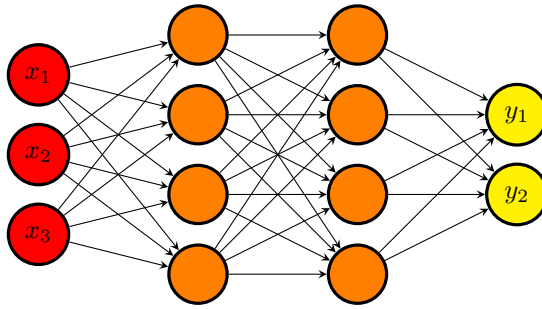


Figure 1: Fully connected neural network with two hidden layers, modeling a mapping from \mathbb{R}^3 to \mathbb{R}^2 .

185 3. Deep reinforcement learning and proximal policy optimization

186 3.1. Neural networks

187 A neural network (NN) is a collection of artificial neurons, i.e., connected computational units
 188 that can be trained to arbitrarily well approximate the mapping function between input and output
 189 spaces. Each connection provides the output of a neuron as an input to another neuron. Each
 190 neuron performs a weighted sum of its inputs, to assign significance to the inputs with regard to the
 191 task the algorithm is trying to learn. It then adds a bias to better represent the part of the output
 192 that is actually independent of the input. Finally, it feeds an activation function that determines
 193 whether and to what extent the computed value should affect the outcome. As sketched in figure 1,
 194 a fully connected network is generally organized into layers, with the neurons of one layer being
 195 connected solely to those of the immediately preceding and following layers. The layer that receives
 196 the external data is the input layer, the layer that produces the outcome is the output layer, and
 197 in between them are zero or more hidden layers.

198 The design of an efficient neural network requires a proper optimization of the weights and
 199 biases, together with a relevant nonlinear activation function. The abundant literature available
 200 on this topic points to a relevant network architecture (e.g., type of network, depth, width of each
 201 layer), finely tuned hyper parameters (i.e., parameters whose value cannot be estimated from data,
 202 e.g., optimizer, learning rate, batch size) and a sufficiently large amount of data to learn from as
 203 being the key ingredients for success; see, e.g., Ref. [49] and the references therein.

204 3.2. Deep reinforcement learning

205 Deep reinforcement learning (DRL) is an advanced branch of machine learning in which deep
 206 neural networks train in solving sequential decision-making problems. It is a natural extension of
 207 reinforcement learning (RL), in which an agent (the neural network) is taught how to behave in
 208 an environment by taking actions and by receiving feedback from it under the form of a reward
 209 (to measure how good or bad the taken action was) and information (to gauge how the action
 210 has affected the environment). This can be formulated as a Markov Decision Process, for which a
 211 typical execution goes as follows (see also figure 2):

- 212 • assume the environment is in state $s_t \in \mathcal{S}$ at iteration t , where \mathcal{S} is a set of states,
- 213 • the agent uses w_t , an observation of the current environment state (and possibly a partial
 214 subset of s_t) to take action $a_t \in \mathcal{A}$, where \mathcal{A} is a set of actions,
- 215 • the environment reacts to the action and transitions from s_t to state $s_{t+1} \in \mathcal{S}$,
- 216 • the agent is fed with a reward $r_t \in \mathcal{R}$, where \mathcal{R} is a set of rewards, and a new observation
 217 w_{t+1} ,

218 This repeats until some termination state is reached, the succession of states and actions defining
 219 a trajectory $\tau = (s_0, a_0, s_1, a_1, \dots)$. In any given state, the objective of the agent is to determine
 220 the action maximizing its cumulative reward over an episode, i.e., over one instance of the scenario

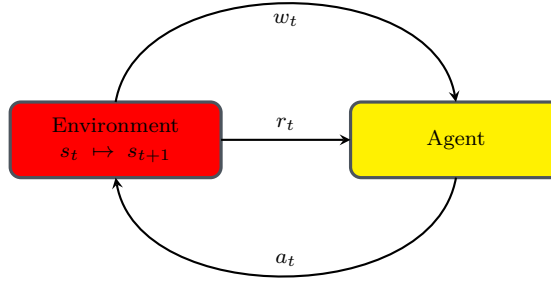


Figure 2: RL agent and its interactions with its environment.

221 in which the agent takes actions. Most often, the quantity of interest is the discounted cumulative
 222 reward along a trajectory defined as

$$R(\tau) = \sum_{t=0}^T \gamma^t r_t, \quad (17)$$

223 where T is the horizon of the trajectory, and $\gamma \in [0, 1]$ is a discount factor that weighs the relative
 224 importance of present and future rewards (the agent being short-sighted in the limit where $\gamma \rightarrow 0$,
 225 since it then cares solely about the first reward, and far-sighted in the limit where $\gamma \rightarrow 1$, since it
 226 then cares equally about all rewards).

227 There exist two main types of RL algorithms, namely model-based methods, in which the
 228 agent tries to build a model of how the environment works to make predictions about what the
 229 next state and reward will be before taking any action, and model-free methods, in which the agent
 230 conversely interacts with the environment without trying to understand it, and are prominent in
 231 the DRL community. Another important distinction to be made within model-free algorithms
 232 is that between value-based methods, in which the agent learns to predict the future reward of
 233 taking an action when provided a given state, then selects the maximum action based on these
 234 estimates, and policy-based methods, in which it optimizes the expected reward of a decision policy
 235 mapping states to actions. Many of the most successful algorithms in DRL (including proximal
 236 policy optimization, whose assessment for flow control and optimization purposes is the primary
 237 motivation for this research) proceed from policy gradient methods, in which gradient ascent is
 238 used to optimize a parameterized policy with respect to the expected return, as further explained
 239 in the next section. The reader interested in a more thorough introduction to the zoology of RL
 240 methods (together with their respective pros and cons) is referred to Ref. [50].

241 3.3. From policy methods to Proximal policy optimization

242 This section intended for the non-specialist reader briefly reviews the basic principles and as-
 243 sumptions of policy gradient methods, together with the various steps taken for improvement.

244
 245 - *Policy methods.* A policy method maximizes the expected discounted cumulative reward of a
 246 decision policy mapping states to actions. It resorts not to a value function, but to a probability
 247 distribution over actions given states, that fully defines the behavior of the agent. Since policies
 248 are most often stochastic, the following notations are introduced:

- 249 • $\pi(s, a)$ is the probability of taking action a in state s under policy π ,
- 250 • $Q^\pi(s, a)$ is the expected value of the return of the policy after taking action a in state s (also
 251 termed state-action value function or Q-function)

$$Q^\pi(s, a) = \mathbb{E}_\pi [R(\tau)|s, a], \quad (18)$$

252 • $V^\pi(s)$ is the expected value of the return of the policy in state s (also termed value function
253 or V-function)

$$V^\pi(s) = \mathbb{E}_\pi [R(\tau)|s]. \quad (19)$$

254 The V and Q functions are therefore such that

$$V^\pi(s) = \sum_a \pi(s, a) Q^\pi(s, a), \quad (20)$$

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

257 - *Policy gradient methods.* A policy gradient method aims at optimizing a parametrized policy
258 π_θ , where θ denotes the free parameters whose value can be learnt from data (as opposed to the
259 hyper parameters). In practice, one defines an objective function based on the expected discounted
260 cumulative reward

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [R(\tau)], \quad (21)$$

261 and seeks the parameterization θ^* maximizing $J(\theta)$, hence such that

$$\theta^* = \arg \max_\theta \mathbb{E}_{\tau \sim \pi_\theta} [R(\tau)], \quad (22)$$

262 which can be done on paper by plugging an estimator of the policy gradient $\nabla_\theta J(\theta)$ into a gradient
263 ascent algorithm. This is no small task as one is looking for the gradient with respect to the policy
264 parameters, in a context where the effects of policy changes on the state distribution are unknown
265 (since modifying the policy will most likely modify the set of visited states, which will in turn affect
266 performance in some indefinite manner). The most commonly used estimator, derived in [50] using
267 the log-probability trick, reads

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^T \nabla_\theta \log(\pi_\theta(s_t, a_t)) R(\tau) \right] \sim \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^T \nabla_\theta \log(\pi_\theta(s_t, a_t)) A^\pi(s, a) \right], \quad (23)$$

268 where the rightmost term is a convenient approximation relying on the advantage function

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s), \quad (24)$$

269 that measures the improvement (if $A > 0$, otherwise the lack thereof) associated with taking action
270 a in state s compared to taking the average over all possible actions. This is because the value
271 function does not depend on θ , so taking it off changes neither the expected value, nor the gradient,
272 but it does reduce the variance, and speeds up the training. Furthermore, when the policy π_θ is
273 represented by a neural network (in which case θ simply denotes the network weights and biases
274 to be optimized), the focus is rather on the policy loss defined as

$$L(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^T \log(\pi_\theta(a_t|s_t)) A^\pi(s, a) \right], \quad (25)$$

275 whose gradient is equal to the (approximated) policy gradient (23) (since the gradient operator acts
276 only on the log-policy term, not on the advantage) and is computed with respect to each weight
277 and bias by the chain rule, one layer at the time, using the back-propagation algorithm [10].
278

279 - *Trust regions.* The performance of policy gradient methods is hurt by the high sensitivity to the
280 learning rate, i.e., the size of the step to be taken in the gradient direction. Indeed, small learning
281 rates are detrimental to learning, but large learning rates can lead to a performance collapse if the
282 agent falls off the cliff and restarts from a poorly performing state with a locally bad policy. This
283 is all the more harmful as the learning rate cannot be tuned locally, meaning that an above average

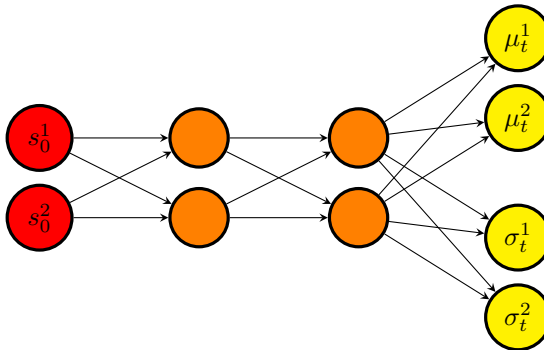


Figure 3: Agent network example used to map states to policy. The input state \mathbf{s}_0 , here of size 2, is mapped to a mean $\boldsymbol{\mu}$ and a standard deviation $\boldsymbol{\sigma}$ vectors, each of size 2. All activation functions are ReLu, except for that of the last layer, which are linear for the $\boldsymbol{\mu}$ output, and softplus for the $\boldsymbol{\sigma}$ output. Orthogonal weights initialization is used throughout the network.

284 learning rate will speed up learning in some regions of the parameter space where the policy loss
 285 is relatively flat, but will possibly trigger an exploding policy update in other regions exhibiting
 286 sharper variations. One way to ensure continuous improvement is by imposing a trust region con-
 287 straint to limit the difference between the current and updated policies, which can be done by
 288 determining first a maximum step size relevant for exploration, then by locating the optimal point
 289 within this trust region. We will not dwell on the intricate details of the many algorithms developed
 290 to solve such trust region optimization problems, e.g., natural policy gradient (NPG [51]), or trust
 291 region policy optimization (TRPO [52]). Suffice it to say that they use the minorize-maximization
 292 algorithm to maximize iteratively a surrogate policy loss (i.e. a lower bound approximating locally
 293 the actual loss at the current policy), but are difficult to implement and can be computationally
 294 expensive, as they rely on an estimate of the second-order gradient of the policy log probability.
 295

296 - *Proximal policy optimization..* Proximal policy optimization (PPO) is another approach with
 297 simple and effective heuristics, that uses a probability ratio between the two policies to maximize
 298 improvement without the risk of performance collapse [22]. The focus here is on the PPO-clip
 299 algorithm¹, that optimizes the surrogate loss

$$L(\theta) = \mathbb{E}_{(s,a) \sim \pi_\theta} \left[\min \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} A^{\pi_\theta}(s, a), g(\epsilon, A^{\pi_\theta}(s, a)) \right) \right], \quad (26)$$

300 where

$$g(\epsilon, A) = \begin{cases} (1 + \epsilon)A & A \geq 0, \\ (1 - \epsilon)A & A < 0, \end{cases} \quad (27)$$

301 and $\epsilon \in [0.1, 0.3]$ is the clipping range, a small hyper parameter defining how far away the new
 302 policy is allowed to go from the old. In practice, a state-action pair yielding a positive advantage
 303 will cause $\pi_\theta(a|s)$ to increase for the action to become more likely. By doing so, if it increases
 304 in a way such that $\pi_\theta(a|s) > (1 + \epsilon)\pi_{\theta_{old}}(a|s)$, the min kicks in (26) and its argument hits a
 305 ceiling of $(1 + \epsilon)A^{\pi_\theta}(s, a)$. Conversely, a state-action pair yielding a negative advantage will cause
 306 $\pi_\theta(a|s)$ to decrease for the action to become less likely. If it increases in a way such that $\pi_\theta(a|s) <$
 307 $(1 - \epsilon)\pi_{\theta_{old}}(a|s)$, the max kicks in and its argument hits a ceiling of $(1 - \epsilon)A^{\pi_\theta}(s, a)$. In neither case
 308 does the new policy has any incentive to step too far away from the current policy, which ensures
 309 that both policies will behave similarly.

310 The strengths of the approach lie in the fact that the clipped surrogate itself is cheap to
 311 compute, and that it needs only an estimate of the first-order gradient of the policy log probability

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.

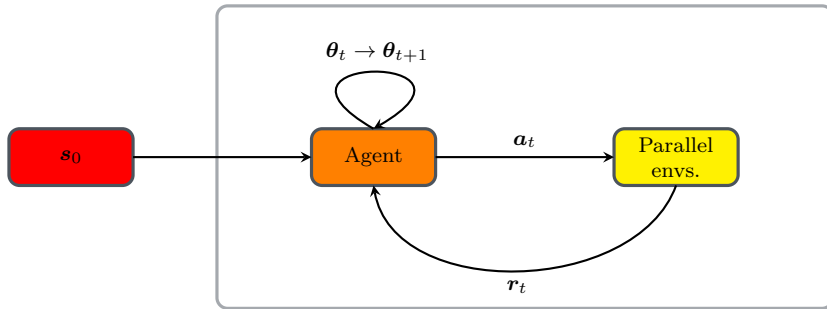


Figure 4: Action loop for single-step PPO. At each episode, the input state s_0 is provided to the agent, which in turn provides n actions to n parallel environments. The latter return n rewards, that evaluate the quality of each action taken. Once all the rewards are collected, an update of the agent parameters is made using the PPO loss (26).

312 (like policy gradient methods). Refinements have been proposed recently, for instance the Trust
 313 region PPO (TRGPPO) that adaptively adjusts the clipping range within the trust region [53], but
 314 classic PPO is generally acknowledged to be one of the most successful DRL method, combining
 315 ease of implementation and tuning, together with state-of-the-art performance across a wide range
 316 of challenging tasks.

317 3.4. Single-step PPO

318 We now come to the single-step PPO method proposed in [26], a “degenerate” version of PPO
 319 in which the agent and the environment get to exchange only one single triplet (s, a, r) per learning
 320 episode. This represents a significant difference with respect to classic Markov-based methods. and
 321 amounts to view the agent as a parameterized mapping f_{θ} from state to action.

322 As sketched in figure 3, the agent is consistently fed with the same input state s_0 (usually a
 323 vector of zeros), and outputs a policy π that can be represented by mean and standard deviation
 324 vectors (denoted by μ and σ , respectively) whose size matches the dimension of the action required
 325 by the environment. The optimization loop is as follows: first, the agent implements a random
 326 mapping f_{θ_0} from s_0 to an initial policy determined by the initialization of the network parameters
 327 θ_0 . At each iteration, a population of actions $\mathbf{a}_t = f_{\theta_t}(s_0)$ is drawn from the current policy, the
 328 reward associated to each set of action is computed and the agent is returned incentives (through
 329 the rewards) to update the free parameters in a way such that the next population of actions
 330 $\mathbf{a}_{t+1} \sim f_{\theta_{t+1}}(s_0)$ will yield higher rewards (see also figure 4). This is done following for the most
 331 part the various steps described in section 3.3, only one seeks in practice the optimal mapping
 332 $f_{\theta_{opt}}$ such that $\mathbf{a}_{opt} \sim f_{\theta_{opt}}(s_0)$, not the optimal set of actions \mathbf{a}_{opt} itself, and the loss is computed
 333 from (26) by substituting a normalized averaged reward for the advantage function.

334 It is worth noticing that an accurate estimation of the policy gradient requires evaluating a large
 335 amount of actions drawn from the current policy, which comes at the expense of computing the same
 336 amount of reward evaluations. One key issue in the context of CFD applications is thus the ability
 337 to distribute a given set of actions to a parallel environment running on large computer clusters,
 338 as we show in the following that the CPU cost of solving the present steady-state optimization
 339 problems ranges from a few tens to several thousand hours.

340 The present workflow relies on the online PPO implementation of Stable Baselines, a toolset of
 341 reinforcement learning algorithms dedicated to the research community and industry [54], for which
 342 a custom OpenAI environment has been designed using the Gym library [55]. The instant reward
 343 r_t used to train the neural network is simply the quantity subjected to optimization (modulo a
 344 plus or minus sign to tackle both maximization and minimization problems). A moving average
 345 reward is also computed on the fly as the sliding average over the 100 latest values of r_t (or the
 346 whole sample if the latter has size smaller than 100). All other relevant hyper parameters are
 347 documented in the next sections, with the exception of the discount factor (since single-step PPO
 348 computes only one single reward per episode).

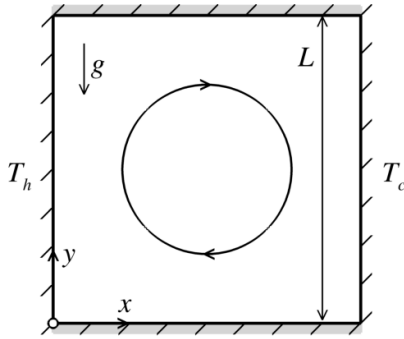


Figure 5: Schematic of the two-dimensional Rayleigh-Bénard set-up.

349 4. Control of natural convection in 2-D closed cavity

350 4.1. Case description

351 We address first the control of natural convection in the two-dimensional differentially heated
 352 square cavity schematically illustrated in figure 5(a). This is a widely studied benchmark system for
 353 thermally-driven flows, relevant in nature and technical applications (e.g., ocean and atmospheric
 354 convection, materials processing, metallurgy), that is thus suitable to validate and compare numerical
 355 solution algorithms while enriching the knowledge base for future projects in this field.
 356 A Cartesian coordinate system is used with origin at the lower-left edge, horizontal x -axis, and
 357 vertical y -axis. The cavity has side L , its top and bottom horizontal walls are perfectly insulated
 358 from the outside, and the vertical sidewalls are isothermal. Namely, the right sidewall is kept at
 359 a constant, homogeneous “cold” temperature T_c , and the left sidewall is entirely controllable via a
 360 constant in time, varying in space “hot” distribution $T_h(y)$ such that

$$\langle T_h \rangle > T_c, \quad (28)$$

361 where the brackets denote the average over space (here over the vertical position along the sidewall).

362 In the following, we neglect radiative heat transfer ($\chi = 0$) and consider a Boussinesq system
 363 driven by buoyancy, hence

$$\boldsymbol{\psi} = \rho_0 \beta (T - T_c) g \mathbf{e}_y, \quad (29)$$

364 where \mathbf{g} is the gravitational acceleration parallel to the sidewalls, β is the thermal expansion
 365 coefficient, and we use the cold sidewall temperature as Boussinesq reference temperature. By
 366 doing so, the pressure featured in the momentum equation (2) and related weak forms must be
 367 understood as the pressure correction representing the deviation from hydrostatic equilibrium. The
 368 governing equations are solved with no-slip conditions $\mathbf{u} = \mathbf{0}$ on $\partial\Omega$ and temperature boundary
 369 conditions

$$\partial_y T(x, 0, t) = \partial_y T(x, L, t) = 0, \quad T(0, y, t) = \langle T_h \rangle + \tilde{T}_h(y), \quad T(L, y, t) = T_c, \quad (30)$$

370 where \tilde{T}_h is a zero-mean (in the sense of the average over space) distribution of hot temperature
 371 fluctuations subjected to optimization, whose magnitude is bounded by some constant ΔT_{max}
 372 according to

$$|\tilde{T}_h(y)| \leq \Delta T_{max}, \quad (31)$$

373 to avoid extreme and nonphysical temperature gradients. All results are made non-dimensional
 374 using the cavity side, the heat conductivity time, and the well-defined, constant in time difference
 375 between the averaged sidewall temperatures. The retained fluid properties yield values of the
 376 Rayleigh and Prandtl numbers

$$\text{Ra} = \frac{g\beta(\langle T_h \rangle - T_c)L^3}{\nu\alpha} = 10^4, \quad \text{Pr} = \frac{\nu}{\alpha} = 0.71, \quad (32)$$

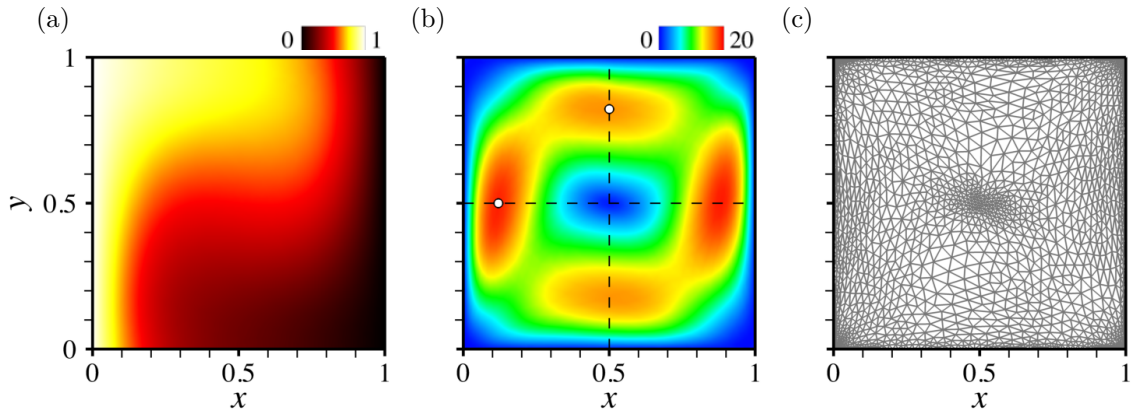


Figure 6: Iso-contours of the uncontrolled steady state (a) temperature and (b) velocity magnitude. (c) Adapted mesh. The circle symbols in (b) mark the positions of the maximum horizontal and vertical velocity along the centerlines reported in table 1.

	Present	Ref. [57]	Ref. [56]	Ref. [58]	Ref. [59]	Ref. [60]
Nu	2.267	2.245	2.238	2.201	2.245	2.245
$\max u(0.5, y)$	16.048	16.179	16,178	–	16.262	16.178
y_{\max}	0.823	0.824	0.823	0.832	0.818	0.827
$\max v(x, 0.5)$	19.067	19.619	19.617	–	19.717	19.633
x_{\max}	0.120	0.121	0.119	0.113	0.119	0.123

Table 1: Comparison of the present numerical results in the absence of control with reference benchmark solutions from the literature.

377 where $\alpha = \lambda/(\rho c_p)$ is the thermal diffusivity.

378 In order to assess the accuracy of the numerical framework, the uncontrolled solution has
379 been computed by performing 60 iterations with time step $\Delta t = 0.5$ to march the initial solution
380 (consisting of zero velocity and uniform temperature, except at the hot sidewall) to steady state.
381 At each time step, an initially isotropic mesh is adapted under the constraint of a fixed number
382 of elements $n_{el} = 4000$ using a multiple-component criterion featuring velocity and temperature,
383 but no level-set. This is because the case is heat transfer but not conjugate heat transfer, as the
384 solid is solely at the boundary $\partial\Omega$ of the computational domain, where either the temperature is
385 known, or the heat flux is zero. It is thus implemented without the IVM and without a level set
386 (although accurate IVM numerical solutions have been obtained in [40] using thick sidewalls with
387 high thermal conductivity). The solution shown in figure 6(a,b) features a centered roll confined
388 by the cavity walls, consistently with the fact that Ra exceeds the critical value $\text{Ra}_c \sim 920$ for the
389 onset of convection (as extrapolated from the near-critical benchmark data in [56]) by one order
390 of magnitude, and heat transfer is thus driven by both conduction and convection. This shows in
391 the Nusselt number, i.e., the non-dimensional temperature gradient averaged over the hot sidewall

$$\text{Nu} = -\langle \partial_x T \rangle, \quad (33)$$

392 whose present value $\text{Nu} = 2.27$ (as computed from 68 points uniformly distributed along the
393 sidewall) exceeds that $\text{Nu} = 1$ of the purely conductive solution, and exhibits excellent agreement
394 with benchmark results from the literature. This is evidenced in table 1 where we also report the
395 magnitude and position of the maximum horizontal velocity u (resp. the vertical velocity v) along
396 the vertical centerline (resp. the horizontal centerline). The corresponding adapted mesh shown in
397 figure 6(c) stresses that all boundary layers are sharply captured via extremely stretched elements,
398 and that the adaptation strategy yields refined meshes near high temperature gradients and close
399 to the side walls. Note however, the mesh refinement is not only along the boundary layers but also
400 close to the recirculation regions near the cavity center, while the elements in-between are coarse
401 and essentially isotropic.

4.2. Control

The question now being raised is whether DRL can be used to find a distribution of temperature fluctuations \tilde{T}_h capable of alleviating convective heat transfer. To do so, we follow [27] and train a DRL agent in selecting piece-wise constant temperature distributions over n_s identical segments, each of which allows only two pre-determined states referred to as hot or cold. This is intended to reduce the complexity and the computational resources, as large/continuous action spaces are known to be challenging for the convergence of RL methods [61, 62]. Simply put, the network action output consists of n_s values $\hat{T}_{hk \in \{1 \dots n_s\}} = \pm \Delta T_{max}$, mapped into the actual fluctuations according to

$$\tilde{T}_{hk} = \frac{\hat{T}_{hk} - \langle \hat{T}_{hk} \rangle}{\max_l \left\{ 1, \frac{|\hat{T}_{hl} - \langle \hat{T}_{hl} \rangle|}{\Delta T_{max}} \right\}}, \quad (34)$$

to fulfill the zero-mean and upper bound constraints.² Ultimately, the agent receives the reward $r_t = -\text{Nu}$ to minimize the space averaged heat flux at the hot sidewall.

All results reported herein are for $\Delta T_{max} = 0.75$ (so the hot temperature varies in the range $[0.25; 1.75]$) and $n_s = 10$ segments, as [27] report that $n_s = 20$ was computationally too demanding for their case, and that $n_s = 5$ yielded poor control efficiency. The agent is a fully-connected network with two hidden layers, each holding 2 neurons. The resolution process uses 8 environments and 2 steps mini-batches to update the network for 32 epochs, with learning rate 5×10^{-3} , and PPO loss clipping range $\epsilon = 0.2$.

4.3. Results

For this case, 120 episodes have been run, each of which follows the exact same procedure as above and performs 60 iterations with time step $\Delta t = 0.5$ to march the zero initial condition to steady state. This represents 960 simulations, each of which is performed on 4 cores and lasts 20s, hence 5h of total CPU cost. We present in figure 7 representative iso-contours of the steady-state temperature and velocity magnitude computed over the course of the optimization. The latter exhibit strong temperature gradients at the hot sidewall, together with a robust steady roll-shaped pattern accompanied by a small corner eddy at the upper-left edge of the cavity, whose size and position depends on the specifics of the temperature distribution. The corresponding meshes are displayed in figure 7(c) to stress the ability of the adaptation procedure to handle well the anisotropy of the solution caused by the intrinsic flow dynamics and the discontinuous boundary conditions.

We show in figure 8 the evolution of the controlled averaged Nusselt number, whose moving average decreases monotonically and reaches a plateau after about 90 episodes, although we notice that sub-optimal distributions keep being explored occasionally. The optimal computed by averaging over the 10 latest episodes (hence the 800 latest instant values) is $\langle \text{Nu} \rangle_{\text{opt}} \sim 0.57$, with variations ± 0.01 computed from the root-mean-square of the moving average over the same interval, which is a simple yet robust criterion to assess qualitatively convergence a posteriori. Interestingly, the optimized Nusselt number is almost twice as small as the purely conductive value ($\text{Nu} = 1$), meaning that the approach successfully alleviates the heat transfer enhancement generated by the onset of convection, although it does not alleviates convection itself, as evidenced by the consistent roll-shaped pattern in figure 9. Similar results are reported in [27], for a different set-up in which the horizontal cavity walls are isothermal and control is applied at the bottom at the cavity (hence a different physics because of buoyancy), albeit with lower numerical and control efficiency since the authors report an optimal Nusselt number $\text{Nu} \sim 1$ using up to 512 DRL environments with learning rate of 2.5×10^{-4} . The reason for such discrepancies probably lies in different ways of achieving and assessing control, as we use single-step PPO to optimize the steady-state Nusselt

²Another possible approach would have been to penalize the reward passed to the DRL for those temperature distributions deemed non-admissible (either because the average temperature is non-zero or the temperature magnitude is beyond the threshold). However, this would have made returning admissible solutions part of the tasks the network is trained on (not to mention that non-zero average temperatures amount to a change in the Rayleigh number), which would likely have slowed down learning substantially.

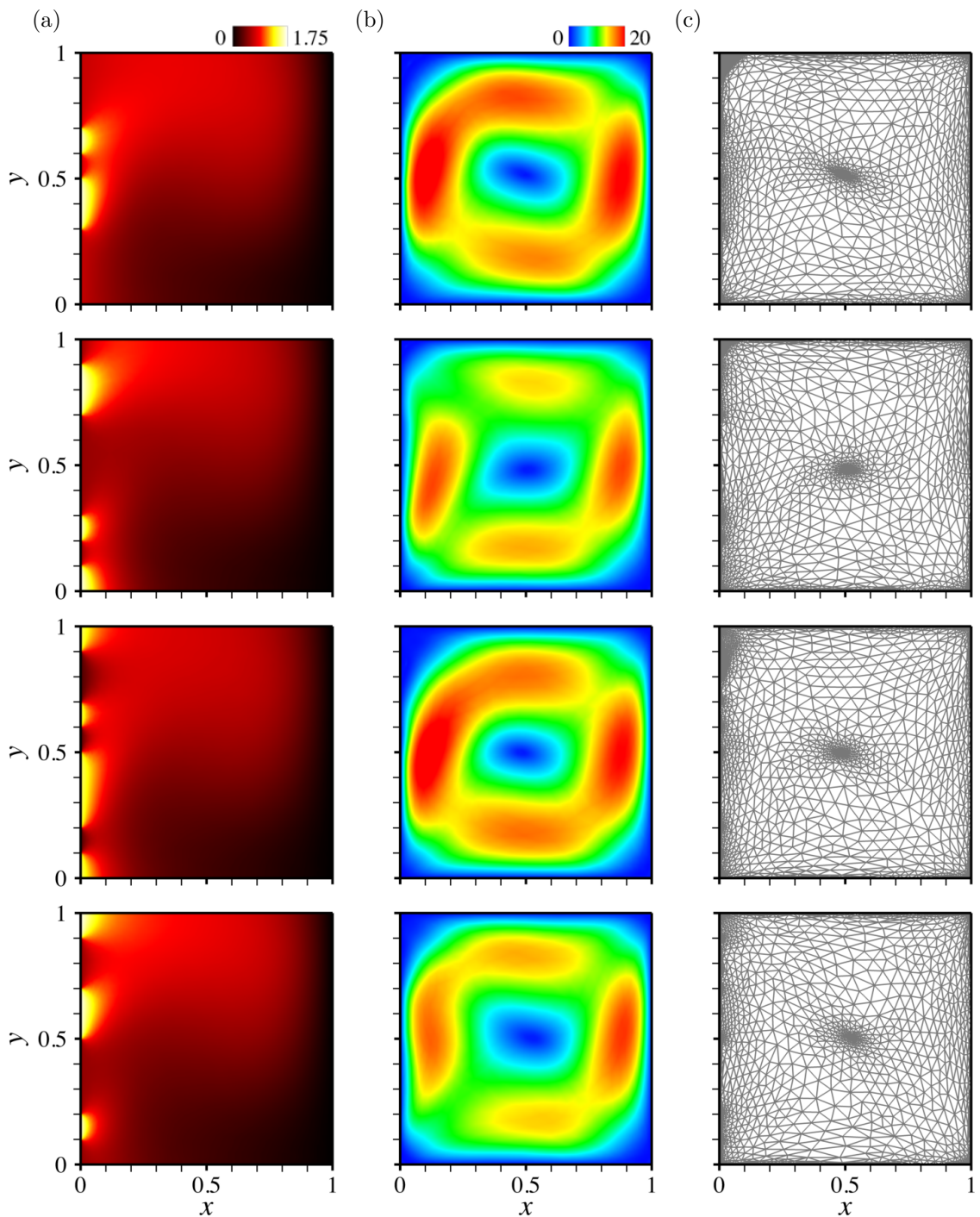


Figure 7: (a,b) Steady-state (a) temperature and (b) velocity magnitude against zero-mean temperature distributed at the left sidewall. (c) Adapted meshes.

446 number via a time-independent control, which requires choosing a sidewall temperature, march-
 447 ing the controlled solution to steady state, then computing the reward. The problem considered
 448 in [27] is more intricate, as classic PPO is used to optimize the reward accumulated over time via
 449 a time-dependent control temperature updated with a certain period scaling with the convection
 450 time in the cavity (the so-determined optimal control being ultimately time-independent for the
 451 considered value of Ra , but truly time-dependent for Rayleigh numbers above $\sim 10^5$).

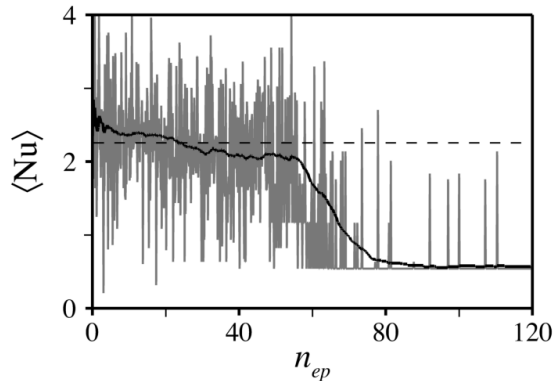


Figure 8: Evolution per learning episode of the instant (in grey) and moving average (in black) Nusselt number. The horizontal dashed line marks the uncontrolled value.

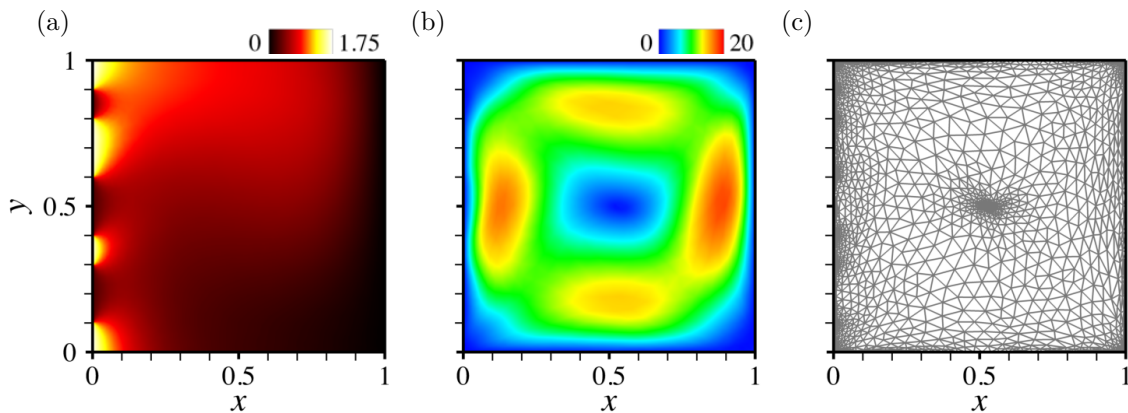


Figure 9: (a,b) Steady-state (a) temperature and (b) velocity magnitude for the optimal zero-mean temperature distribution. (c) Adapted mesh.

452 5. Control of forced convection in 2-D open cavity

453 5.1. Case description

454 This second test case addresses the control of actual conjugate heat transfer in a model setup
 455 for the cooling of a hot solid by impingement of a fluid; see figure 10(a). A Cartesian coordinate
 456 system is used with origin at the center of mass of the solid, horizontal x -axis, and vertical y -axis.
 457 The solid has rectangular shape with height h and aspect ratio 2:1, and is initially at the hot
 458 temperature T_h . It is fixed at the center of a rectangular cavity with height H and aspect ratio
 459 4:1, whose walls are isothermal and kept at temperature T_w . The top cavity side is flush with n_j
 460 identical holes of width e_i whose distribution is subjected to optimization, each of which models
 461 the exit plane of an injector blowing cold air at velocity V_i and temperature T_c , and is identified by
 462 the horizontal position of its center $x_{k \in \{1 \dots n_j\}}$. In order to allow releasing hot air from the cavity,
 463 the sidewalls are blown with two identical exhaust areas of height e_o , identified by the vertical
 464 position of their center $(e_0 - H)/2$.

465 For this case, both buoyancy and radiative heat transfer are neglected (hence, $\psi = \mathbf{0}$ and
 466 $\chi = 0$), meaning that temperature evolves as a passive scalar, similar to the mass fraction of a
 467 reactant in a chemical reaction. All relevant parameters are provided in Table 2, including the
 468 material properties used to model the composite fluid, that yield fluid values of the Reynolds and
 469 Prandtl numbers

$$\text{Re} = \frac{\rho V_i e}{\mu} = 200, \quad \text{Pr} = 2. \quad (35)$$

470 Note the very high value of the ratio of the solid to fluid viscosities, that ensures that the velocity
 471 is zero in the solid domain and that the no-slip interface condition is satisfied. By doing so, the

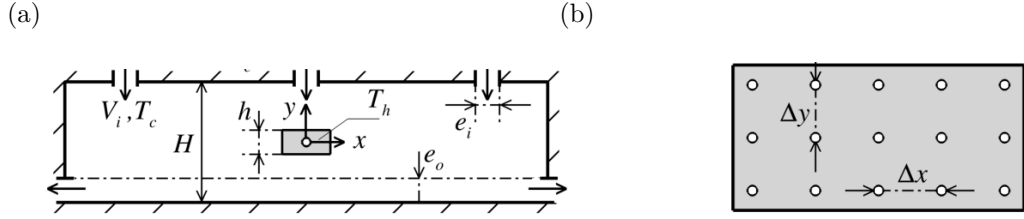


Figure 10: (a) Schematic of the 2-D forced convection set-up. (b) Sensors positions in the solid domain.

H	h	e_i	e_o	V_i	T_w	T_c	T_h
1	0.2	0.2	0.2	1	10	10	150
				μ	ρ	λ	c_p
		fluid		0.001	1	0.5	1000
		solid		1000	100	15	300

Table 2: Numerical parameters used in the 2-D forced convection problem. All values in SI units, with the exception of temperatures given in Celsius.

convective terms drop out in the energy equation, that reduces to the pure conduction equation for the solid. The governing equations are solved with no-slip isothermal conditions $\mathbf{u} = \mathbf{0}$ and $T = T_w$ on $\partial\Omega$, except at the injection exit planes ($\mathbf{u} = -V_i \mathbf{e}_y$, $T = T_c$), and at the exhaust areas, where a zero-pressure outflow condition is imposed ($p = \partial_x u = \partial_x T = 0$). No thermal condition is imposed at the interface, where heat exchange is implicitly driven by the difference in the individual material properties.

5.2. Control

The quantity being optimized is the distribution of the injectors center positions $x_{k \in \{1 \dots n_j\}}$, each of which is forced to sit in an interval $[x_k^-, x_k^+]$ whose edge values are determined beforehand or recomputed on the fly (depending on the control strategy), and bounded according to

$$|x_k^\pm| \leq x_m, \quad (36)$$

where we set $x_m = 2H - 0.75e_i$ to avoid numerical issues at the upper cavity edges. The network action output therefore consists of n_j values $\hat{x} \in [-1; 1]$, mapped into the actual positions according to

$$x_k = \frac{x_k^+(\hat{x}_k + 1) - x_k^-(\hat{x}_k - 1)}{2}. \quad (37)$$

In order to compute the reward passed to the DRL, we distribute uniformly 15 probes in the solid domain, into $n_x = 5$ columns and $n_y = 3$ rows with resolutions $\Delta x = 0.09$ and $\Delta y = 0.075$, respectively; see figure 10(b). Selected tests have been carried out to check that the outcome of the learning process does not change using $n_y = 5$ rows of $n_x = 5$ probes (not shown here). The magnitude of the tangential heat flux is estimated by averaging the norm of the temperature gradient over all columns and rows, i.e., i -th column (resp. the j -th row) as

$$\langle \|\nabla_{\parallel} T\| \rangle_i = \frac{2}{n_y - 1} \left| \sum_{j \neq 0} \text{sgn}(j) \|\nabla T\|_{ij} \right|, \quad \langle \|\nabla_{\parallel} T\| \rangle_j = \frac{2}{n_x - 1} \left| \sum_{i \neq 0} \text{sgn}(i) \|\nabla T\|_{ij} \right|, \quad (38)$$

where subscripts i , j and ij denote quantities evaluated at $x = i\Delta x$, $y = j\Delta y$ and $(x, y) = (i\Delta x, j\Delta y)$, respectively, and symmetrical numbering is used for the center probe to sit at the intersection of the zero-th column and row. The numerical reward $r_t = -\langle \|\nabla_{\parallel} T\| \rangle$ fed to the DRL agent deduces ultimately by averaging over all rows and columns, to give

$$\langle \|\nabla_{\parallel} T\| \rangle = \frac{1}{n_x + n_y} \sum_{i,j} \langle \|\nabla_{\parallel} T\| \rangle_i + \langle \|\nabla_{\parallel} T\| \rangle_j, \quad (39)$$

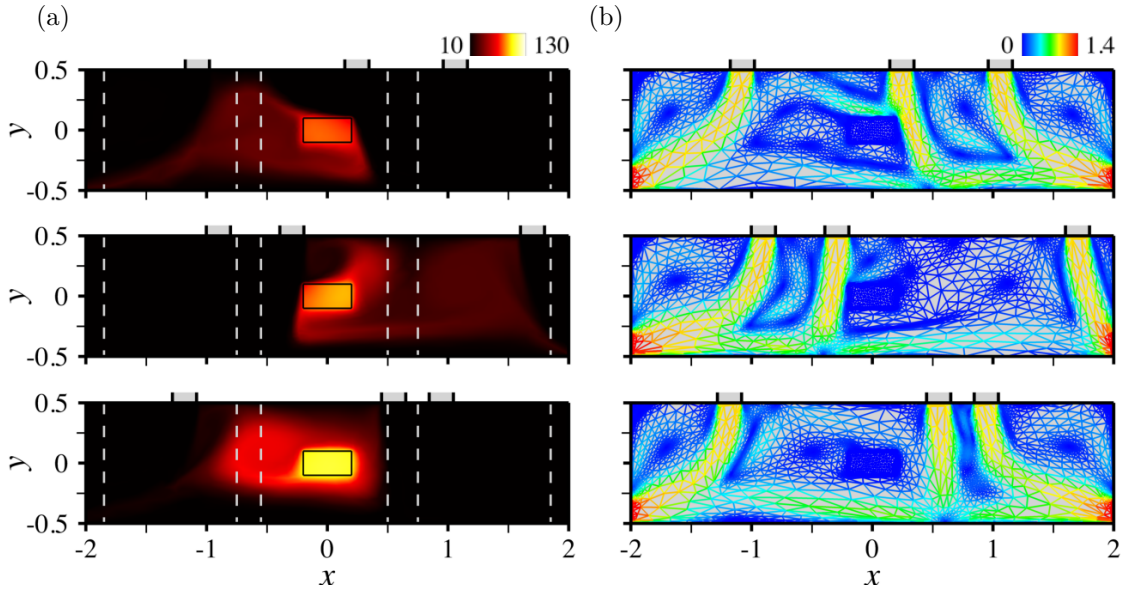


Figure 11: (a) Steady-state temperature against arrangements of 3 injectors, with admissible values under the fixed domain decomposition strategy S_1 delimited by the dashed lines. (b) Adapted meshes colored by the magnitude of velocity.

495 which especially yields $r_t = 0$ for a perfectly homogeneous cooling.

496 All results reported in the following are for $n_j = 3$ injectors. The agent is a fully-connected
 497 network with two hidden layers, each holding 2 neurons. The resolution process uses 8 environments
 498 and 2 steps mini-batches to update the network for 32 epochs, with learning rate set to 5×10^{-3} ,
 499 and PPO loss clipping range to $\epsilon = 0.3$.

500 5.3. Results

501 5.3.1. Fixed domain decomposition strategy

502 We consider first the so-called fixed domain decomposition strategy S_1 in which the top cavity
 503 wall is split into n_j equal subdomains, and each injector is forced to sit in a different subdomain.
 504 The edge values for the position x_k of the k -th injector read

$$x_k^- = -x_m + (k-1) \frac{2x_m + e_i}{n_j}, \quad x_k^+ = x_k^- + \frac{2x_m - (n_j - 1)e_i}{n_j}. \quad (40)$$

505 It can be checked that $x_k^- = x_{k-1}^+ + e_i$, so it is possible to end up with two side-by-side injectors,
 506 which is numerically equivalent to having $n_j - 1$ injectors, $n_j - 2$ of width e_i plus one of width $2e_i$.
 507 For this case, 60 episodes have been run, each of which performs 1500 iterations with time step
 508 $\Delta t = 0.1$ to march the same initial condition (consisting of zero velocity and uniform temperature,
 509 except in the solid domain) to steady state, using the level set, velocity and temperature as multiple-
 510 component criterion to adapt the mesh (initially pre-adapted using the sole level set) every 5
 511 time steps under the constraint of a fixed number of elements $n_{el} = 15000$. This represents 480
 512 simulations, each of which is performed on 8 cores and lasts 10mn, hence 80h of total CPU cost.

513 It is out of the scope of this work to analyze in details the many flow patterns that develop when
 514 the blown fluid travels through the cavity. Suffice it to say that the outcome depends dramatically
 515 on the injectors arrangement, and features complex rebound phenomena (either fluid/solid, when
 516 a jet impinges on the cavity walls or on the workpiece itself, or fluid/fluid, when a deflected jet
 517 meets the crossflow of another jet), leading to the formation of multiple recirculations varying in
 518 number, position and size. Several such cases are illustrated in figure 11 via iso-contours of the
 519 steady-state temperature distributions, together with the corresponding adapted meshes colored by
 520 the magnitude of velocity to illustrate the ability of the numerical framework to capture accurately
 521 all boundary layers and shear regions via extremely stretched elements.

522 One point worth being mentioned is that the individual position signals are best suited to draw
 523 robust quantitative conclusion, as there is noise in the reward signal shown in figure 12(a). The

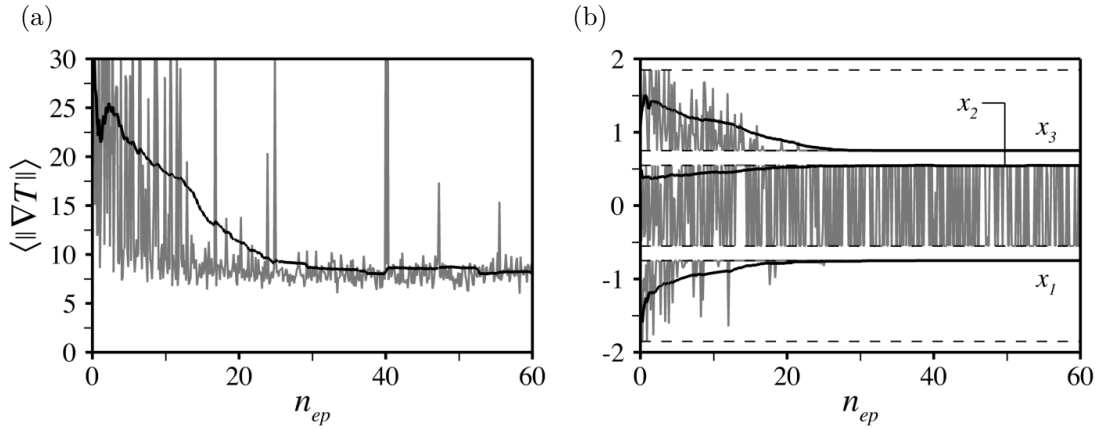


Figure 12: (a) Evolution per learning episode of the instant (in grey) and moving average (in black) rewards under the fixed domain decomposition strategy S_1 . (b) Same as (a) for the injectors center positions, with admissible values delimited by the dashed lines.

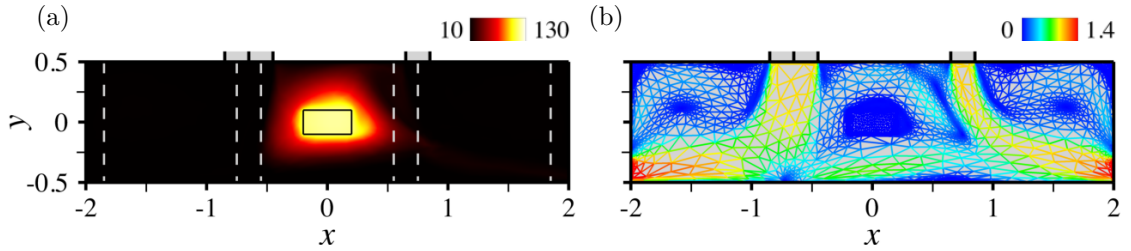


Figure 13: Same as figure 11 for the optimal arrangement of 3 injectors under the fixed domain decomposition strategy S_1 .

524 issue, we believe, is twofold: on the one hand, the reward is approximated from pointwise tem-
525 perature data (similar to experimental measurements) that are more sensitive to small numerical
526 errors (e.g., the interpolation error at the probes position) than an integral quantity. On the other
527 hand, the mesh adaptation procedure is not a deterministic process, as the outcome depends on
528 the processors and number of processors used, and any initial difference propagates over the course
529 of the simulation because the meshes keep being adapted dynamically. In return, two exact same
530 control parameters can thus yield different rewards on behalf of different interpolation errors at
531 the probes position. This likely slows down learning and convergence, but we show in figure 12(b)
532 that the moving average distribution does converge to an optimal arrangement after roughly 25
533 episodes. The latter consists of an injector at the right-end of the left subdomain ($x_{1\text{opt}} = -0.75$)
534 and two side-by-side injectors sitting astride the center and right subdomains ($x_{2\text{opt}} = 0.55$ and
535 $x_{3\text{opt}} = 0.75$), that enclose the workpiece in a double-cell recirculation; see figure 13. These values
536 have been computed by averaging the instant positions of each injector over the 10 latest episodes,
537 with variations ± 0.002 computed from the root-mean-square of the moving average over the same
538 interval, a procedure that will be used consistently to assess convergence for all cases reported in
539 the following. The efficiency of the control itself is estimated by computing the magnitude of tan-
540 gential heat flux averaged over the same interval, found to be $\langle \|\nabla_{\parallel} T \rangle_{\text{opt}} \sim 8.3$. Note, the position
541 $x_{2\text{opt}}$ is actually obtained by averaging the absolute value of the instant position x_2 (although the
542 true, signed value is depicted in the figure), which is because the center injector keeps oscillating
543 between two end positions ± 0.55 on behalf of reflectional symmetry with respect to the vertical
544 centerline.

545 5.3.2. Follow-up strategy

546 We consider now the so-called follow-up strategy S_2 , in which all injectors are distributed
547 sequentially the ones with respect to the others. The corresponding edge values

$$x_1^- = -x_m, \quad x_1^+ = x_m - (n_j - 1)e_i, \quad (41)$$

$$x_k^- = x_{k-1}^+ + e_i, \quad x_k^+ = x_m - (n_j - k)e_i, \quad (42)$$

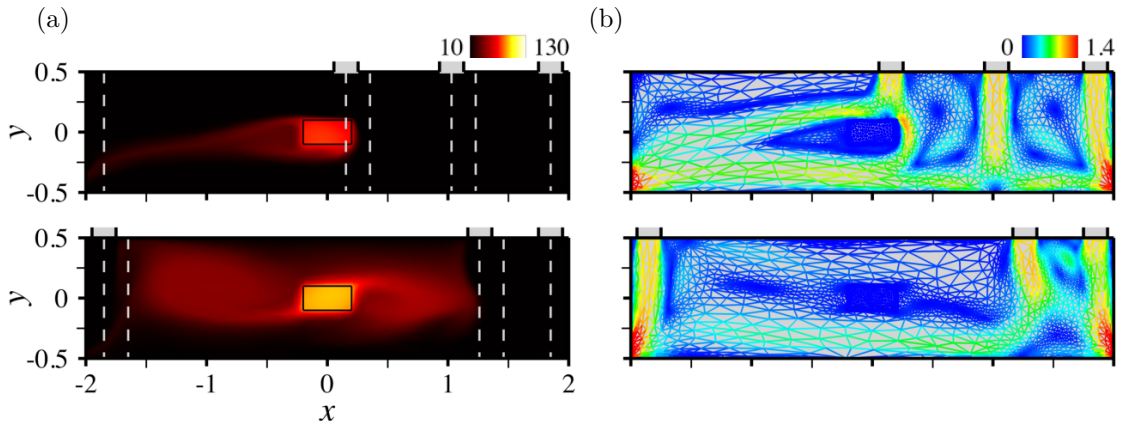


Figure 14: (a) Steady-state temperature against arrangements of 3 injectors, with admissible values under the follow-up strategy S_2 delimited by the dashed lines. (b) Adapted meshes colored by the magnitude of velocity.

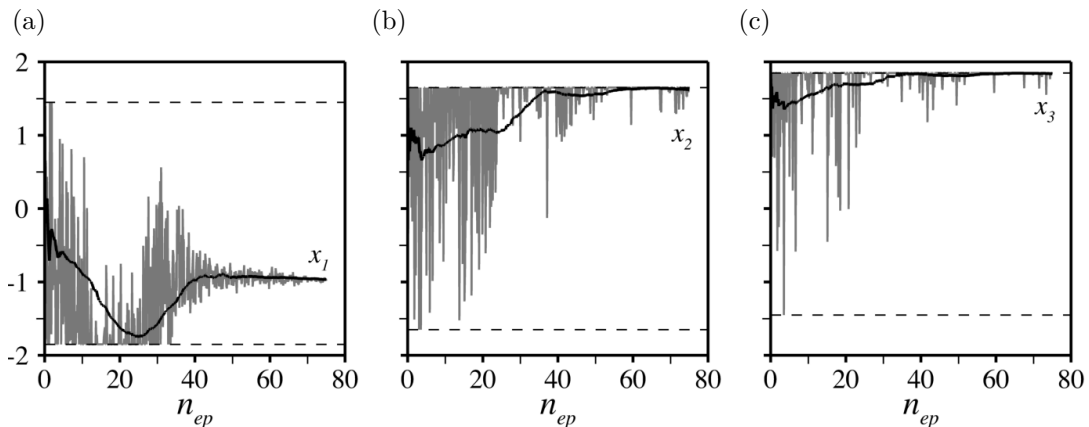


Figure 15: Evolution per learning episode of the instant (in grey) and moving average (in black) injectors center positions under the follow-up strategy S_2 , with admissible values delimited by the dashed lines.

548 readily express that the k -th injector is forced to sit between the $k - 1$ -th one and the upper-right
549 cavity edge while leaving enough space to distribute the remaining $n_j - k$ injectors, which increases
550 the size of the control parameter space while again leaving the possibility for side-by-side injectors
551 (since $x_k^- = x_{k-1}^+ + e_i$ by construction). 75 episodes have been run for this case following the
552 exact same procedure as above, i.e., marching the zero initial condition in time up to $t = 150$ with
553 $\Delta t = 0.1$, hence 600 simulations, each of which is performed on 8 cores and lasts 10mn, hence 100h
554 of total CPU cost.

555 When it comes to the computed flow patterns, the results closely resemble those obtained
556 under the previous fixed domain decomposition strategy, although figure 14 exhibits increased
557 dissymmetry when two or more injectors move simultaneously to the same side of the cavity. We
558 show in figure 15 that the moving average distribution converges after roughly 60 episodes, with
559 the optimal arrangement consisting of one injector roughly midway between the left cavity sidewall
560 and the workpiece ($x_{1\text{opt}} = -0.96$), and two side-by-side injectors at the right end of the cavity
561 ($x_{2\text{opt}} = 1.65$ and $x_{3\text{opt}} = 1.85$). The variations over the same interval are by ± 0.006 ; see also
562 figure 16 for the corresponding flow pattern. Convergence here is much slower than under S_1 , as the
563 search for an optimal is complicated by the fact that all injector positions are interdependent the
564 ones on the others and it is up to the network to figure out exactly how. Another contingent matter
565 is that the agent initially spans a fraction of the control parameter space because the large values
566 of x_1 considered limit the space available to distribute the other two injectors. This is all the more
567 so as such configurations turn to be far from optimality, for instance the magnitude of tangential
568 heat flux is $\langle \|\nabla_{\parallel} T\| \rangle \sim 41.3$ for $x_1 = 1.45$, $x_2 = 1.65$ and $x_3 = 1.85$, but $\langle \|\nabla_{\parallel} T\| \rangle_{\text{opt}} \sim 6.3$ at
569 optimality. The latter value is smaller than the optimal achieved under S_1 , consistently with the
570 fact that all positions spanned under S_1 are admissible under S_2 , hence the S_1 optimal is expected

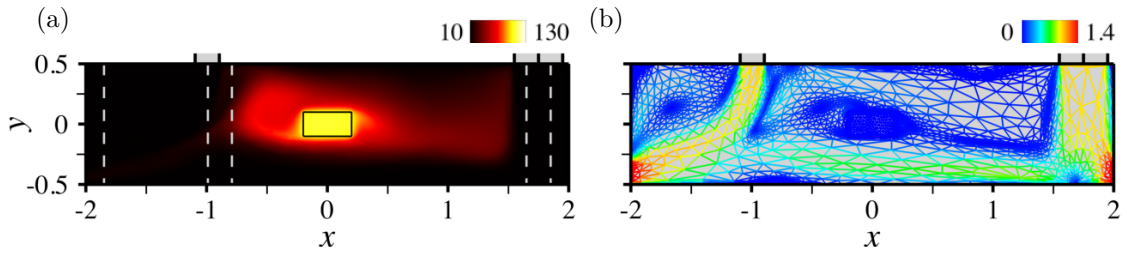


Figure 16: Same as figure 14 for the optimal arrangement of 3 injectors under the follow-up strategy S_2 .

571 to be a S_2 sub-optimal.

572 5.3.3. Free strategy

573 We examine now a third strategy S_3 referred to as the free strategy, in which all injectors are
 574 independent and free to move along the top cavity wall. The edge values for the position x_k of the
 575 k -th injector read

$$x_k^- = -x_m, \quad x_k^+ = x_m, \quad (43)$$

576 so two injectors can end up side-by-side and even overlapping one another if $|x_l - x_m| < e_i$. If
 577 so, we implement a single injector of width $e_i + |x_l - x_m|$ and maintain the blowing velocity (not
 578 the flow rate) for the purpose of automating the set-up design process, meaning that having n_j
 579 injectors, two of which overlap exactly (i.e., $|x_l - x_m| = 0$) is rigorously equivalent to having $n_j - 1$
 580 injectors. 60 episodes have been run for this case following the exact same procedure as above.

581 All flow patterns are reminiscent of those obtained under the previous fix decomposition S_1
 582 and follow-up S_2 strategies, even when two injectors overlap; see figure 17. Other than that, we
 583 show in figures 18 that the moving average distribution converges to an optimal consisting of two
 584 injectors almost perfectly overlapping one another at the left end of the cavity ($x_{1\text{opt}} = -1.85$
 585 and $x_{2\text{opt}} = -1.82$), and a third injector at the right end of the cavity ($x_{3\text{opt}} = 1.85$). The
 586 variations over the same interval are by ± 0.007 , and the associated flow pattern shown in figure 19
 587 is symmetrical and features two large recirculations on either side of the workpiece. Convergence
 588 occurs after roughly 40 episodes, i.e., faster than under S_2 (consistently with the fact that there
 589 is no need to learn anymore about how the network outputs depend the ones on the others) but
 590 slower than under S_1 (consistently with the fact that the size of the control parameter space has
 591 increased substantially). It is worth noticing that the system is invariant by permutations of the
 592 network outputs, meaning that there exist $2^{n_j} - 2$ distributions (hence 6 for $n_j = 3$) associated with
 593 the same reward. Nonetheless, a single optimal is selected, which is essentially fortuitous since the
 594 agent does not learn about symmetries under the optimization process (otherwise S_1 would have
 595 similarly selected a single optimal). The magnitude of tangential heat flux is $\langle \|\nabla_{\parallel} T\| \rangle_{\text{opt}} \sim 11.2$ at
 596 optimality, i.e., larger than that achieved under S_2 . This can seem surprising at first, because all
 597 positions spanned under S_2 are admissible under S_3 , and the S_2 optimal is thus expected to be a
 598 S_3 sub-optimal. However, the argument does not hold here because the overlap in the S_3 optimal
 599 reduces the flow rate to that of a two-injectors set-up, so the comparison should be with the S_2
 600 optimal with $n_j = 2$.

601 5.3.4. Inverse strategy

602 Finally, we propose here to make the most of the numerical framework flexibility to solve a
 603 different optimization problem consisting in selecting first an injector distribution, then in finding
 604 which position x_0 of the solid center of mass minimizes the magnitude of the tangential heat flux
 605 defined by (38)-(39). The so-called inverse strategy S_4 considered herein features two injectors at
 606 each end of the cavity ($x_1 = -1.85$ and $x_2 = 1.85$), identical to the optimal arrangement of 3
 607 injectors under the free strategy S_3 . The center of mass can take any value in $[-x_{0m}; x_{0m}]$ where
 608 we set $x_{0m} = 2(H - h)$ to avoid numerical issues at the sidewalls. The same coordinate system
 609 as above is used, but with reference frame attached to the cavity, not the moving solid (hence all
 610 results obtained under the previous strategies pertain to $x_0 = 0$ in the new system).

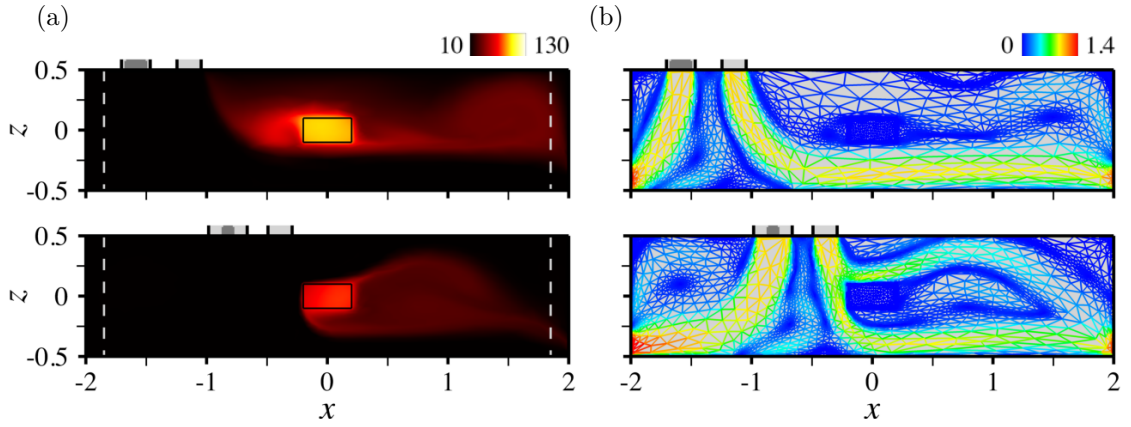


Figure 17: (a) Steady-state temperature against arrangements of 3 injectors, with admissible values under the free strategy S_3 delimited by the dashed lines and overlaps marked by the dark grey shade. (b) Adapted meshes colored by the magnitude of velocity.

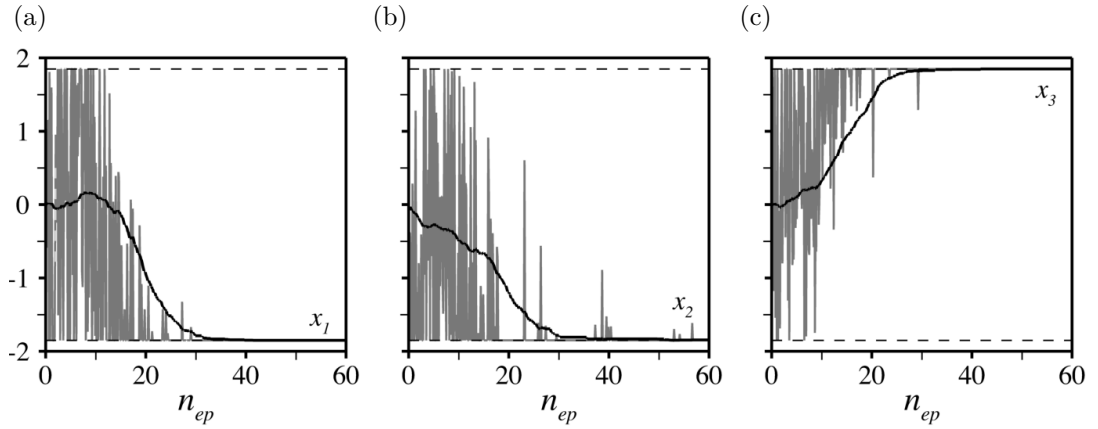


Figure 18: Evolution per learning episode of the instant (in grey) and moving average (in black) injectors center positions under the free strategy S_3 , with admissible values delimited by the dashed lines.

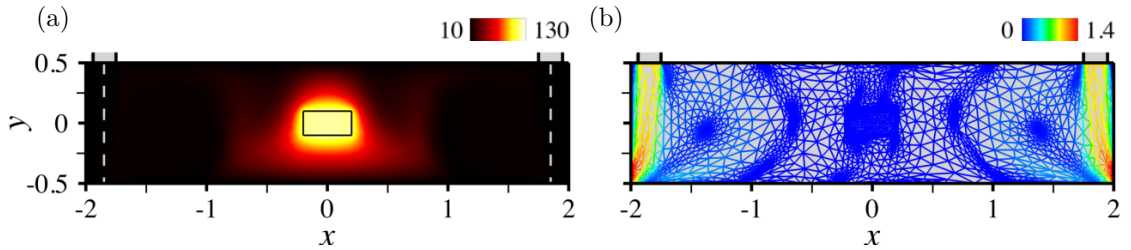


Figure 19: Same as figure 17 for the optimal arrangement of 3 injectors under the free strategy S_3 .

611 A total of 60 episodes have been run for this case using the exact same DRL agent, the only
 612 difference being in the network action output, now made up of a single value $\hat{x}_0 \in [-1; 1]$, mapped
 613 into the actual position using

$$x_0 = x_{0m} \hat{x}_0. \quad (44)$$

614 A large variety of flow patterns is obtained by doing so, that closely resemble those computed
 615 under the previous strategies, only the outcome is now also altered by the width of the gap be-
 616 tween the cavity sidewalls and the workpiece, as illustrated in figure 20. We show in figure 21
 617 that the position of the solid center of mass converges to an optimal position $x_{0\text{opt}} = 0.42$ (the
 618 variations over the same interval being by ± 0.005), the associated magnitude of tangential heat
 619 flux $\langle \|\nabla_{\parallel} T\| \rangle_{\text{opt}} \sim 4.1$, being smaller than that achieved under S_3 using a centered workpiece.
 620 The fact that the optimal position is offset from the horizontal centerline is a little surprising at
 621 first, because intuition suggests that the simplest way to achieve homogeneous heat transfer is

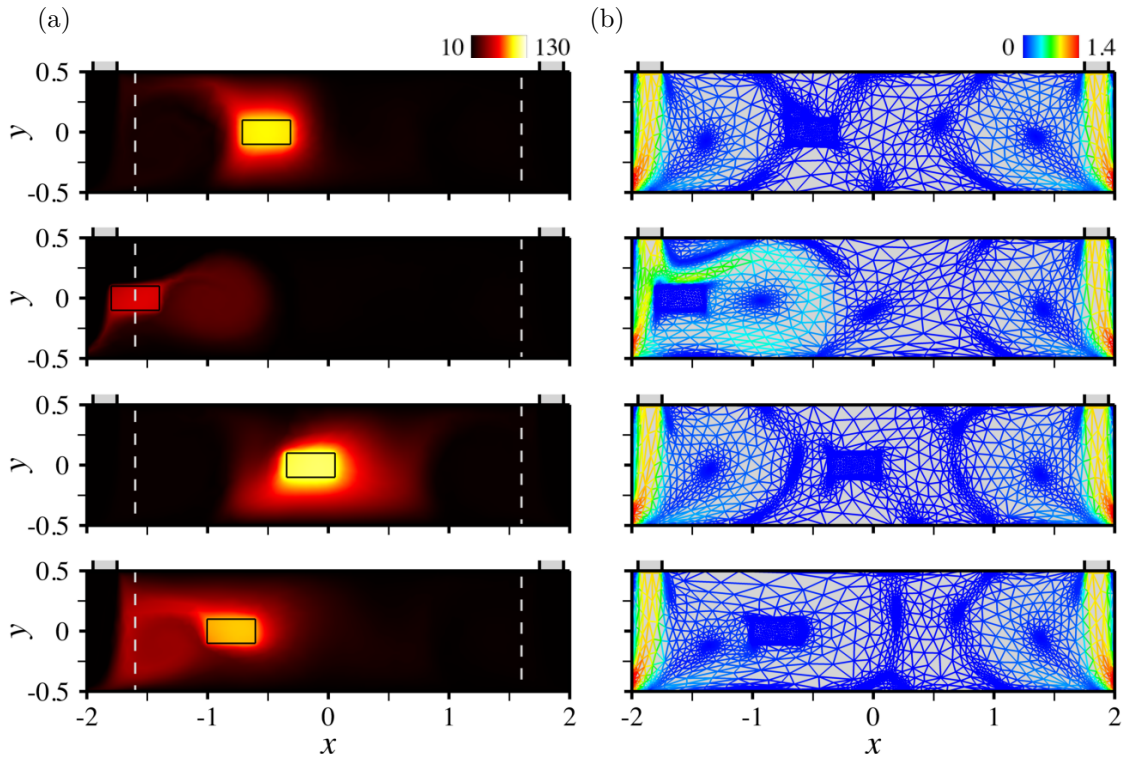


Figure 20: (a) Steady-state temperature against solid center of mass position, with admissible domains under the inverse strategy S_4 marked by the dashed lines. (b) Adapted meshes colored by the norm of velocity.

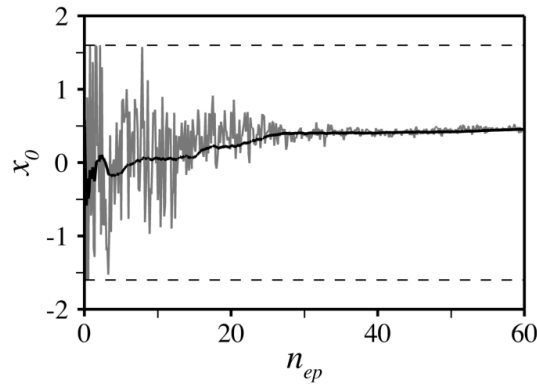


Figure 21: Evolution per learning episode of the instant (in grey) and moving average (in black) center of mass positions under the inverse strategy S_4 , with admissible values delimited by the dashed lines.

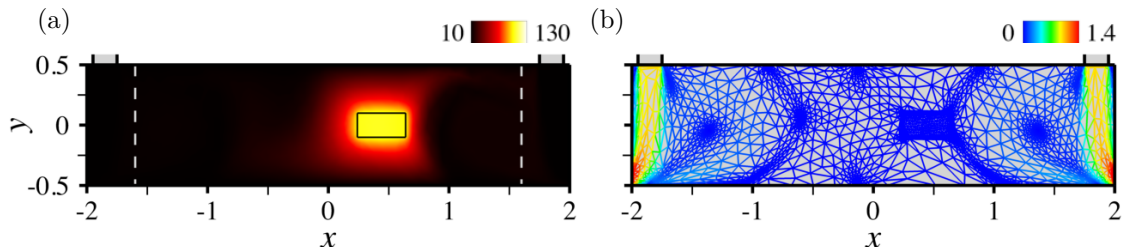


Figure 22: Same as figure 20 for the optimal center of mass position under the inverse strategy S_4 .

622 by having injectors symmetrically distributed with respect to the vertical centerline. Nonetheless,
 623 examining carefully the norm of the temperature gradient in the solid domain shows that $x_0 = 0$
 624 achieves close to perfect horizontal symmetry but vertical asymmetry, owing to the formation of
 625 two large-scale, small velocity end vortices entraining heat laterally downwards; see figure 23(a).

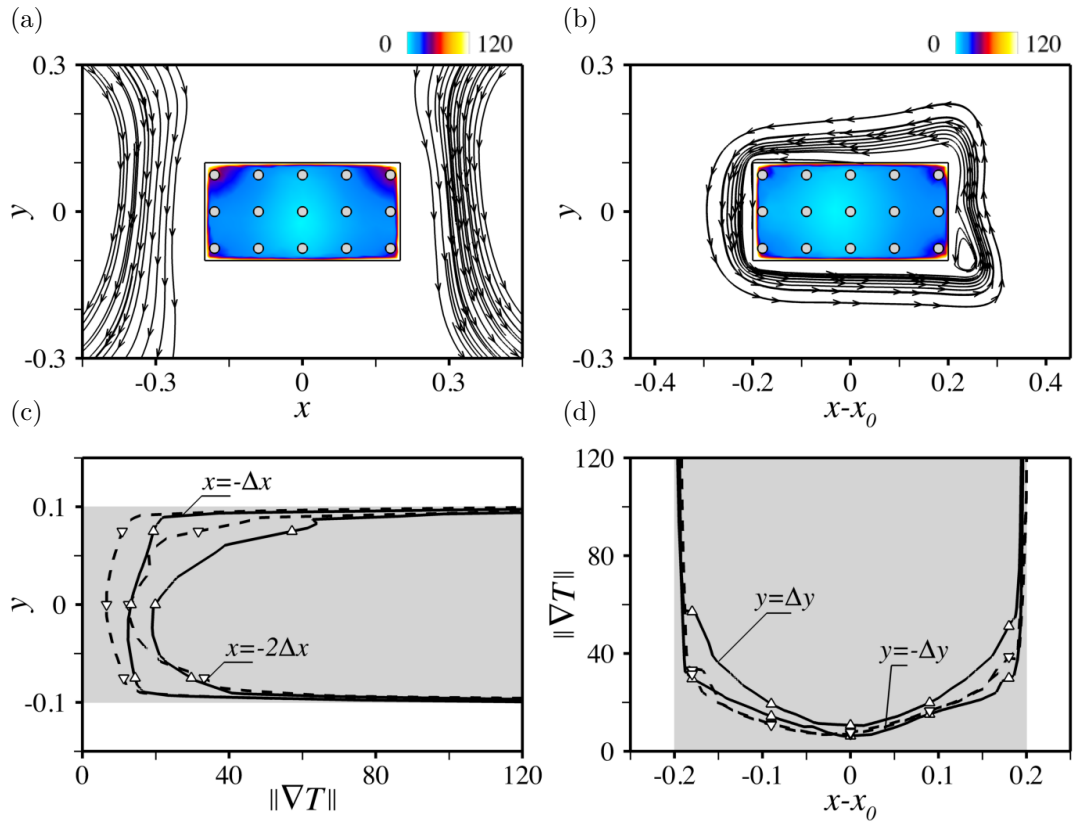


Figure 23: (a,b) Norm of the temperature gradient in the solid domain with superimposed streamlines of the underlying velocity field, as computed for (a) $x_0 = 0$, and (b) $x_{0opt} = 0.45$, i.e., the optimal position selected under the inverse strategy S_4 . (c) Cuts along the two leftmost columns of probes. The solid and dashed lines refer to $x_0 = 0$ and $x_{0opt} = 0.45$, respectively, and the symbols mark the probe values. (d) Same as (c) for cuts along the lower and upper rows of columns.

626 Conversely, for $x \sim x_{0opt}$, the workpiece is almost at the core of the closest recirculation region,
 627 hence the surrounding fluid particles have small velocities and wrap almost perfectly around its
 628 surface, as illustrated in figure 23(b). This restores excellent vertical symmetry, as evidenced by
 629 relevant cuts along the two leftmost columns of probes in figure 23(c), and along the lower and
 630 upper rows in figure 23(d), which explains the improved reward.

631 5.4. Discussion

632 Figure 24 reproduces the optimal temperature distributions computed under the various strate-
 633 gies considered above. For benchmarking purposes, we also provide in table 3 relevant convergence
 634 data computed over the 10 latest episodes. To recap, the most homogeneous cooling is achieved
 635 under the follow-up strategy S_2 , but the DRL agent seems more easily trained under the fixed
 636 decomposition domain strategy S_1 and the free strategy S_3 . Another interesting point is the ex-
 637 tent to which the workpiece is actually cooled, for which S_2 seems more relevant, on behalf of the
 638 dissymmetry in the left and right flow rates that creates order one velocities at the bottom of the
 639 cavity. This stresses S_2 as a possible compromise to achieve efficient *and* homogeneous cooling,
 640 although a true optimal with this regards can be computed rigorously by applying the same ap-
 641 proach to compound functionals weighing, e.g., the magnitude of the tangential heat flux and the
 642 solid center temperature (which we defer to future work).

643 These results provide a basis for future self-assessment of the method and identifies potential
 644 for improvement regarding the convergence efficiency. The approach can certainly benefit from
 645 a fine tuning of the reward computation, as having sufficient spatial resolution on the relevant
 646 state of the system is an obvious requirement to allow a successful control. Adjusting the trade-off
 647 between exploration and exploitation is also worth consideration to better handle the existence of

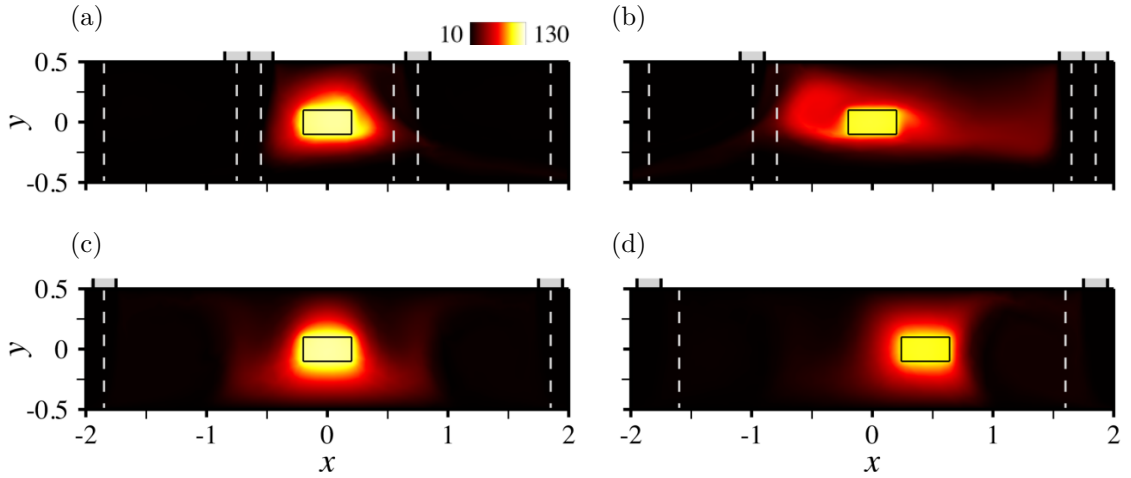


Figure 24: (a-c) Optimal arrangements of 3 injectors under the (a) fixed decomposition domain strategy S_1 , (b) follow-up strategy S_2 and (c) free strategy S_3 . (d) Optimal position of the workpiece under the inverse strategy S_4 .

	n_j	n_{ep}	x_0	x_1	x_2	x_3	$\langle \ \nabla_{\parallel} T\ \rangle$
S_1	3	60	0	-0.75	± 0.55	0.75	8.3
S_2	3	75	0	-0.96	1.65	1.85	6.3
S_3	3	60	0	-1.85	-1.82	1.85	11.2
S_4	2	60	0.42	-1.85	1.85	-	4.1

Table 3: Numerical data for the optimal arrangements computed under strategies S_{1-4} . All values computed by averaging the instant signal over the 10 latest learning episodes.

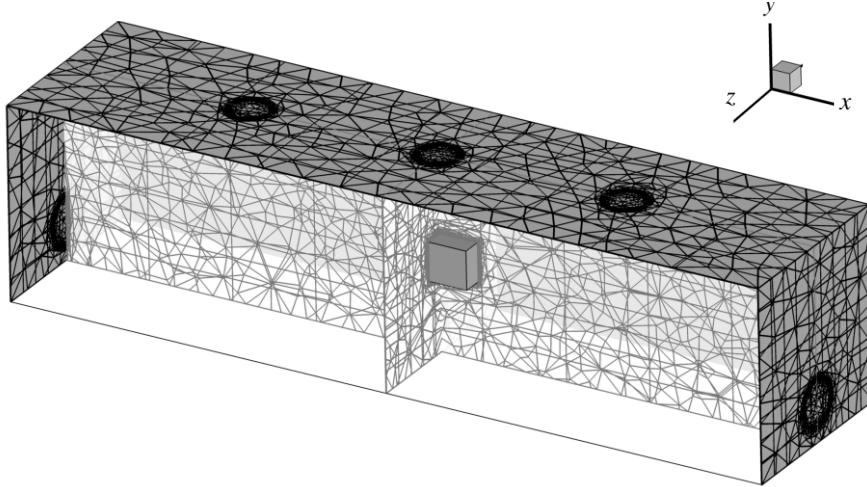


Figure 25: Schematic of the 3-D forced convection set-up.

648 multiple global optima (whether they stem from symmetries of from the topology of the reward
 649 itself) which could be done using non-normal probability density functions.

650 6. Extension to 3-D forced convection

651 6.1. Case description

652 The model cooling set up considered in section 5 is extended here to 3-D to assess the extent to
 653 which the approach carries over to three-dimensional conjugate heat transfer. The main differences
 654 between 2-D and 3-D are as follows: a Cartesian coordinate system is used with origin at the center
 655 of mass of the solid, horizontal x -axis, vertical y -axis, and the z -axis completes the direct triad; see
 656 figure 25. The solid is a rectangular prism with aspect ratio 2:1:1, and is fixed at the center of a

H	h	d_i	d_o	δ_o	V_i	T_w	T_c	T_h
1	0.2	0.2	0.24	0.16	1	10	10	150
			μ	ρ	λ	c_p		
		fluid	0.01	1	0.5	1000		
		solid	1000	100	15	300		

Table 4: Numerical parameters used in the 3-D forced convection problem. All values in SI units, with the exception of temperatures given in Celsius.

657 rectangular cavity with height H and aspect ratio 4:1:1. We consider n_j circular-shaped injectors
658 with diameter d_i , whose exit planes are forced to be symmetrical with respect to $z = 0$, hence each
659 injector is identified by the horizontal position of its center $x_{k \in \{1 \dots n_j\}}$. We also use circular-shaped
660 exhaust areas with diameter d_o , offset by a distance δ_o from the bottom of the cavity, and whose
661 exit planes are also symmetrical with respect to $z = 0$, hence each exhaust area is identified by the
662 vertical position of its center $(d_o + \delta_o - H)/2$. The governing equations are solved with the exact
663 same boundary conditions as in section 5. All parameters are provided in Table 4, including the
664 material properties used to model the composite fluid, that yield fluid values of the Reynolds and
665 Prandtl numbers

$$\text{Re} = \frac{\rho V_i d_i}{\mu} = 20, \quad \text{Pr} = 20. \quad (45)$$

666 6.2. Control strategy

667 We keep here the same control objective and compute the reward fed to the DRL from 45
668 probes arranged symmetrically into $n_z = 3$ transverse layers with resolution $\Delta z = 0.075$, each
669 of which distributes uniformly 15 probes into $n_x = 5$ columns and $n_y = 3$ rows with resolutions
670 $\Delta x = 0.09$ and $\Delta y = 0.075$. In practice, the 3-D reward is simply the average over z of the 2-D
671 reward defined in section 5, hence $r_t = -\langle \|\nabla_{\parallel} T\| \rangle$ with

$$\langle \|\nabla_{\parallel} T\| \rangle = \frac{1}{(n_x + n_y)n_z} \sum_{i,j,k} \langle \|\nabla_{\parallel} T\| \rangle_{ik} + \langle \|\nabla_{\parallel} T\| \rangle_{jk}, \quad (46)$$

672 with

$$\langle \|\nabla_{\parallel} T\| \rangle_{ik} = \frac{2}{n_y - 1} \left| \sum_{j \neq 0} \text{sgn}(j) \|\nabla T\|_{ijk} \right|, \quad \langle \|\nabla_{\parallel} T\| \rangle_{jk} = \frac{2}{n_x - 1} \left| \sum_{i \neq 0} \text{sgn}(i) \|\nabla T\|_{ijk} \right|, \quad (47)$$

673 and the subscripts ik , jk and ijk denote quantities evaluated at $(x, z) = (i\Delta x, k\Delta z)$, $(y, z) =$
674 $(j\Delta y, k\Delta z)$ and $(x, y, z) = (i\Delta x, j\Delta y, k\Delta z)$, respectively.

675 All results reported in the following are for $n_j = 3$ injectors. The edge values needed to map the
676 network action output into the actual injectors positions deduce straightforwardly from (40)-(43)
677 substituting the diameter d_i of the 3-D injectors for the length e_i of the 2-D injectors. The same
678 DRL agent is used, that consists of two hidden layers, each holding 2 neurons, and the resolution
679 process uses 8 environments and 2 steps mini-batches to update the network for 32 epochs. Each
680 environment performs 1250 iterations with time step $\Delta t = 0.1$ to march the same initial condition
681 (consisting of zero velocity and uniform temperature, except in the solid domain) to steady state,
682 using the level set, velocity and temperature as multiple-component criterion to adapt the mesh
683 (initially pre-adapted using the sole level set) every 10 time steps under the constraint of a fixed
684 number of elements $n_{el} = 120000$. This is likely insufficient to claim true numerical accuracy, but
685 given the numerical cost (320 3-D simulations per strategy, each of which is performed on 8 cores
686 and lasts 2h30, hence 800h of total CPU cost), we believe this is a reasonable compromise to assess
687 feasibility while producing qualitative results to build on.

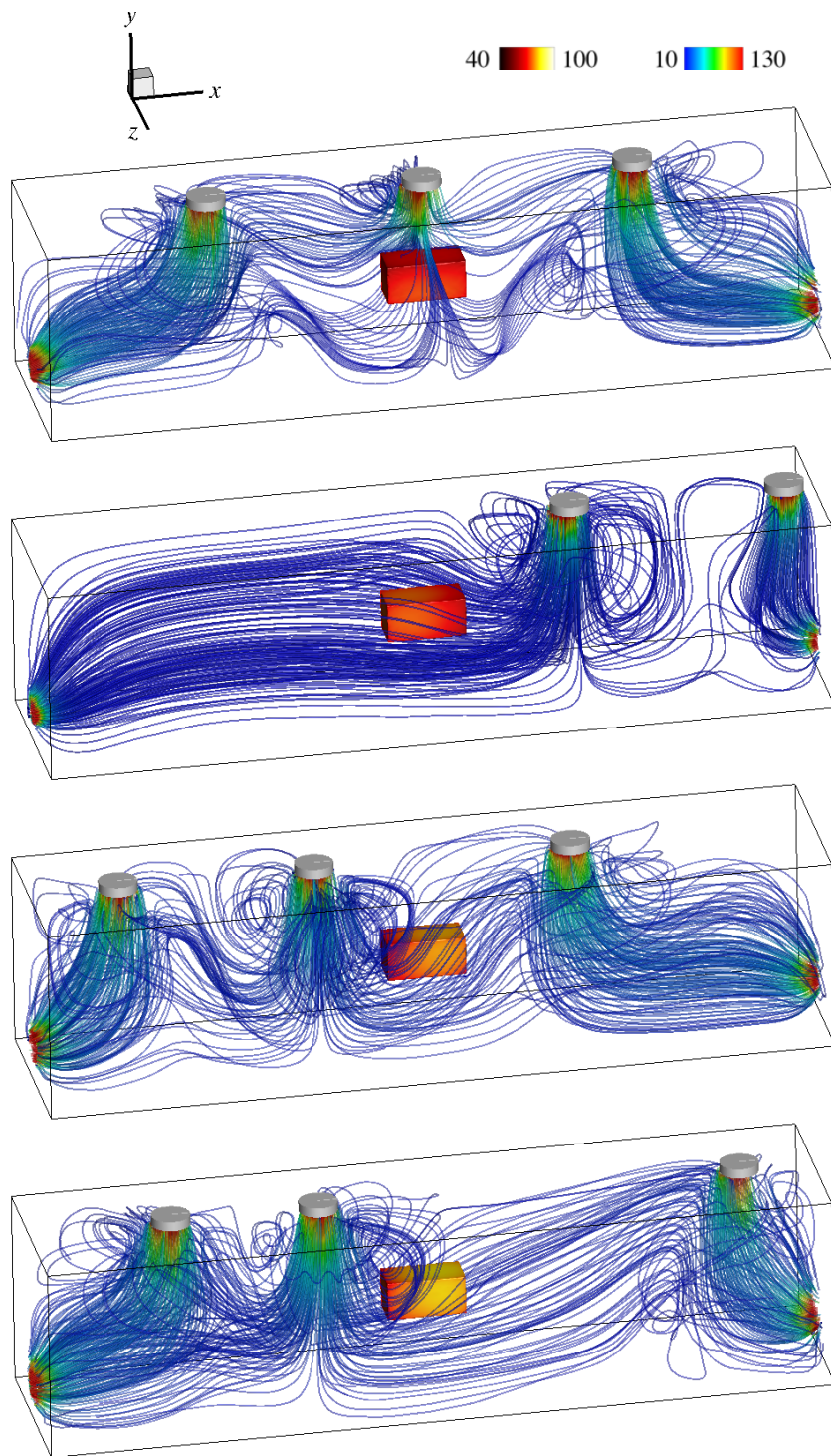


Figure 26: Representative steady-state temperature distributions at the solid/fluid interface together with 3-D streamlines colored by the magnitude of velocity.

688 *6.3. Results*

689 Only the fixed domain decomposition S_1 strategy (in which the top cavity wall is split into
 690 n_j equal subdomains and each injector is forced to sit in a different subdomain) and the free S_3
 691 strategy (in which the injectors are entirely independent and free to move along the top cavity
 692 wall) are considered here to save computational resources, as learning has been seen to be slower
 693 in 2-D under the follow-up S_2 strategy.

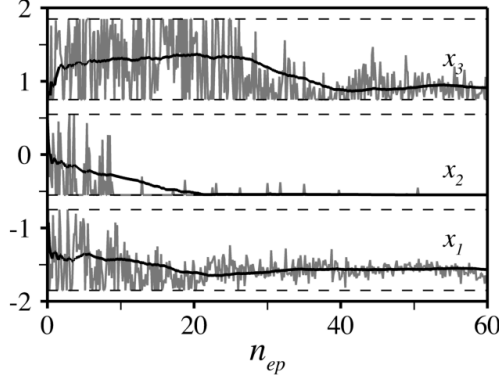


Figure 27: Evolution per learning episode of the instant (in grey) and moving average (in black) injectors center positions under the three-dimensional fixed domain decomposition strategy S_1 , with admissible values delimited by the dashed lines.

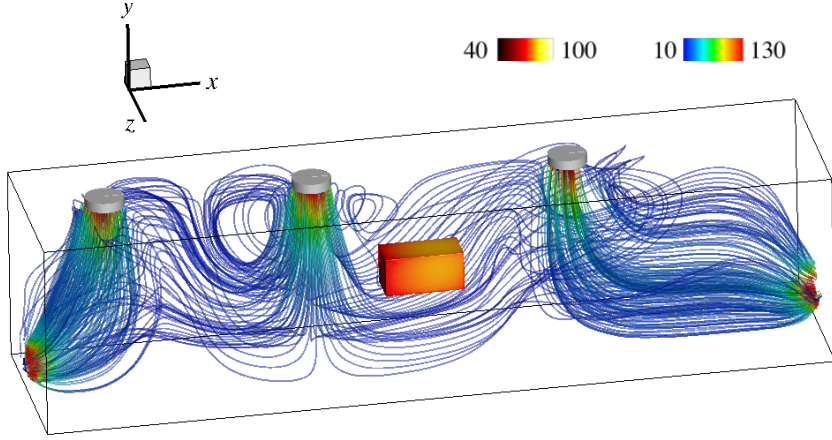


Figure 28: Optimal 3 injector arrangement under the three-dimensional fixed decomposition domain strategy S_1 .

694 A total of 60 episodes have been run under the fixed domain decomposition strategy S_1 . Several
695 representative flow patterns computed over the course of optimization are shown in figure 26 via
696 iso-contours of the steady-state temperature at the fluid-solid interface and 3-D streamlines colored
697 by the magnitude of velocity, to put special emphasis on transverse inhomogeneities and display
698 the increased degree of complexity due to the formation of large-scale horseshoe vortices wrapped
699 around the nozzle jets. We show in figure 27 that the distribution slowly converges to an optimal
700 arrangement consisting of one injector at the left end of the left subdomain ($x_{1\text{opt}} = -1.63$),
701 another one at the left end of the center subdomain ($x_{2\text{opt}} = -0.55$), and a third one at the
702 left end of the right subdomain ($x_{3\text{opt}} = 0.87$), as has been determined by averaging the instant
703 positions of each injector over the latest 10 learning episodes, with variations by roughly ± 0.04
704 computed from the root-mean-square of the moving average over the same interval. This is larger
705 by one order of magnitude than the variations reported in 2-D, as the agent keeps exploring slightly
706 sub-optimal positions of the lateral injectors, which likely simply reflects the challenging nature
707 of performing three-dimensional optimal control. The 3-D S_1 optimal somehow resemble its 2-D
708 counterpart, namely the center injector is at the exact same position, while the lateral injectors
709 (especially the leftmost one) have been pushed towards the cavity sidewalls. The associated flow
710 pattern is reported in figure 28. The associated optimal reward computed over the same interval
711 is $\langle \|\nabla_{\parallel} T\| \rangle_{\text{opt}} \sim 19.5$, i.e. twice as large than in 2-D, although it is difficult to compare further
712 because of the difference in the Reynolds and Prandtl number.

713 Another 40 episodes have been run under the free strategy S_3 , for which the results are almost
714 identical to their 2-D counterparts, as the distribution converges in figure 29 to an optimal ar-
715 rangement consisting of two overlapping injectors at the left end of the cavity ($x_{1\text{opt}} = -1.83$ and
716 $x_{2\text{opt}} = -1.82$), and a third injector at the right end ($x_{3\text{opt}} = 1.83$), with variations by with ± 0.01

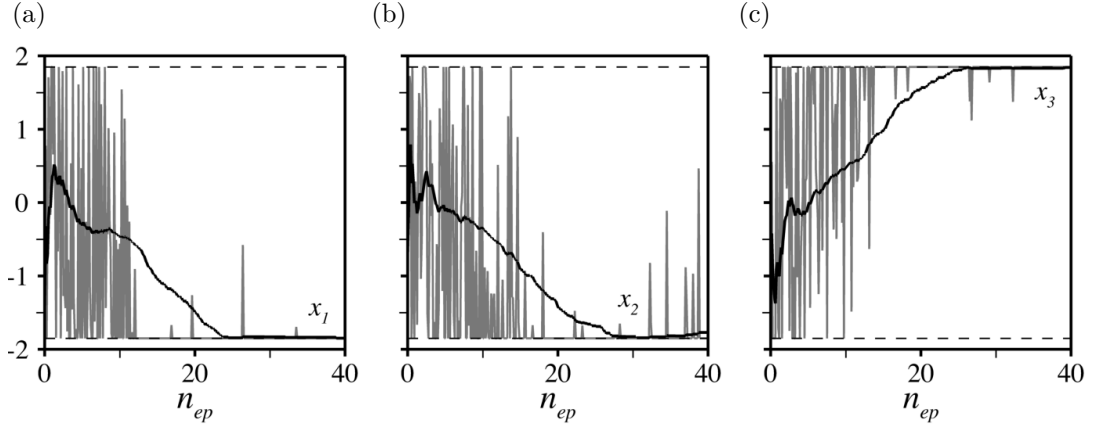


Figure 29: Evolution per learning episode of the instant (in grey) and moving average (in black) injectors center positions under the three-dimensional free strategy S_3 , with admissible values delimited by the dashed lines.

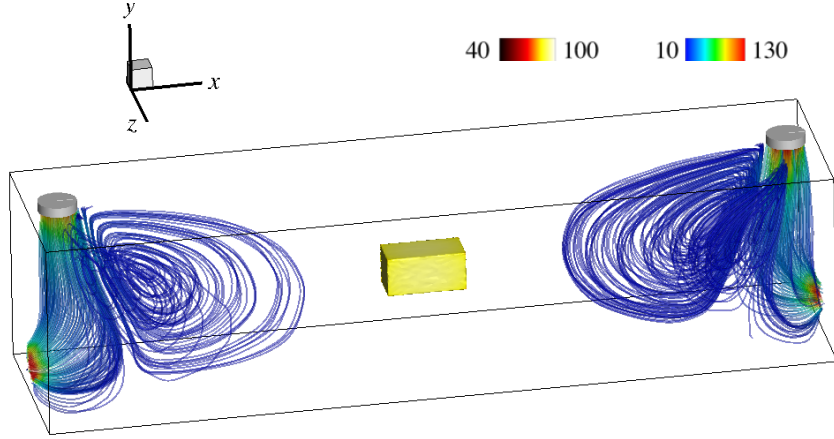


Figure 30: Optimal 3 injector arrangement under the three-dimensional free strategy S_3 .

	n_j	n_{ep}	x_0	x_1	x_2	x_3	$\langle \ \nabla_{\parallel} T\ \rangle$
S_1	3	60	0	-1.63	-0.55	0.87	19.5
S_3	3	40	0	-1.83	-1.82	1.83	4.7

Table 5: Numerical data for the optimal arrangements computed in three-dimensions under strategies S_1 and S_3 . All values computed by averaging the instant signal over the 10 latest learning episodes.

717 for the lateral injectors, but ± 0.03 for the center injector, for which the agent keeps occasionally
718 exploring sub-optimal positions. The corresponding flow pattern shown in figure 30 is thus again
719 symmetrical with two large, 3-D recirculations on either side of the workpiece. The associated
720 optimal reward computed over the same interval is $\langle \|\nabla_{\parallel} T\| \rangle_{\text{opt}} \sim 4.7$ substantially smaller than
721 that achieved under the 3-D S_1 strategy, which again demonstrates the feasibility to improve per-
722 formances by allowing overlaps. All relevant numerical data are reported in table 5 for the sake of
723 completeness.

724 7. Conclusion

725 This research assesses the ability of deep reinforcement learning (DRL) to guide the optimization
726 and control of conjugate heat transfer systems. It combines a novel, “degenerate” version of the
727 proximal policy optimization (PPO) algorithm, that trains a neural network in optimizing the
728 system only once per learning episode, and an in-house stabilized finite elements environment
729 combining variational multiscale (VMS) modeling of the governing equations, immerse volume
730 method, and multi-component anisotropic mesh adaptation, that computes the numerical reward
731 fed to the neural network. The approach is shown capable of optimizing two- and three dimensional

732 systems exhibiting natural and forced convection dominated heat transfer. It is especially applied
733 to identify optimal distribution of injectors and optimal position of a hot workpiece to reduce
734 inhomogeneous impingement cooling under various control strategies.

735 Fluid dynamicists have just begun to gauge the relevance of deep reinforcement learning tech-
736 niques to assist the design of optimal control strategies. This research weighs in on this issue and
737 adds values to the shallow literature on this subject by showing that the proposed single-step PPO
738 holds a high potential as a reliable, go-to black-box optimizer for complex conjugate heat transfer
739 problems. Future research in this field should aim at consolidating the acquired knowledge, at
740 improving the computational efficiency (by a fine-tuning of the hyper parameters, or using pre-
741 trained deep learning models) and at boosting the convergence capabilities (via coupling with a
742 surrogate model trained on-the-fly, or using non-normal probability density functions, or modifying
743 the balance between exploration and exploitation, as the PPO loss is specifically built to avoid a
744 performance collapse by preventing large updates of the policy). The design of relevant numerical
745 rewards under partial state information is another issue that deserves consideration.

746 Scope is another key ingredient to pushing forward the state of the art. A short-term objec-
747 tive would be to quickly tackle more complex test cases exhibiting time-dependent dynamics and
748 turbulence, which the numerical framework is perfectly suited to do via a combination of Reynolds-
749 averaged Navier–Stokes modeling [63, 64] and second-order, semi-implicit time discretization [65].
750 We believe that this will highlight even more clearly the relevance of the methodology, as it is
751 speculated in [30] that DRL should be able to handle chaotic systems without suffering from
752 the shortcomings and limitations of the adjoint method, and [27] consistently report that DRL
753 outperforms a canonical linear proportional-derivative controller in controlling turbulent natural
754 convection. The long-term objective would be to enrich the description of the test cases using
755 multi-physics modeling (e.g., radiative heat transfer, phase transformation), to pave the way to-
756 ward flexible, ready-to-use control of industrially relevant applications, such as thermal comfort
757 for building design or manufacturing processes.

758 Acknowledgements

759 This work is supported by the Carnot M.I.N.E.S. Institute through the M.I.N.D.S. project.

760 References

761 References

- 762 [1] A. Jameson, Aerodynamic design via control theory, *J. Sci. Comput.* 3 (1998) 233–260.
- 763 [2] M. D. Gunzburger, *Perspectives in flow control and optimization*, SIAM, Philadelphia, 2002.
- 764 [3] T. Bewley, Flow control : new challenges for a new renaissance, *Prog. Aerosp. Sci.* 37 (2001)
765 21–58.
- 766 [4] K. Momose, K. Abe, H. Kimoto, Reverse computation of forced convection heat transfer for
767 optimal control of thermal boundary conditions, *Heat Tran. Asian Res.* 33 (2004) 161–174.
- 768 [5] A. Belmiloudi, Robin-type boundary control problems for the nonlinear Boussinesq type equa-
769 tions, *J. Math. Anal. Appl.* 273 (2002) 428–456.
- 770 [6] G. Bärwolff, M. Hinze, Optimization of semiconductor melts, *Z. Angew. Math. Mech.* 86
771 (2006) 423–437.
- 772 [7] J. Boldrini, E. Fernández-Cara, M. Rojas-Medar, An optimal control problem for a generalized
773 Boussinesq model: The time dependent case, *Rev. Mat. Comput.* 20 (2007) 339–366.
- 774 [8] H. Karkaba, T. Dbouk, C. Habchi, S. Russeil, T. Lemenand, D. Bougeard, Multi objective
775 optimization of vortex generators for heat transfer enhancement using large design space
776 exploration, *Chem. Eng. Process.* (2020 (accepted)).

- 777 [9] P. Meliga, J.-M. Chomaz, Global modes in a confined impinging jet: application to heat
778 transfer and control, *Theor. Comput. Fluid Dyn.* 25 (1) (2011) 179–193.
- 779 [10] D. Rumelhart, G. Hinton, R. Williams, Learning representations by back-propagating errors,
780 *Nature* 323 (1986) 533–536.
- 781 [11] J. Kober, J. A. Bagnell, J. Peters, Reinforcement learning in robotics: A survey, *The International
782 Journal of Robotics Research* 32 (11) (2013) 1238–1274.
- 783 [12] V. Mnih, K. Kavukcuoglu, D. Silver, R. A.A., V. J., M. Bellemare, A. Graves, M. Riedmiller,
784 A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King,
785 D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforce-
786 ment learning, *Nature* 518 (2015) 7540.
- 787 [13] G. E. Hinton, A. Krizhevsky, I. Sutskever, Imagenet classification with deep convolutional
788 neural networks, *Advances in neural information processing systems* 25 (2012) 1106–1114.
- 789 [14] B. Lusch, J. N. Kutz, S. L. Brunton, Deep learning for universal linear embeddings of nonlinear
790 dynamics, *Nature communications* 9 (1) (2018) 1–10.
- 791 [15] M. Raissi, A. Yazdani, G. E. Karniadakis, Hidden Fluid Mechanics: A Navier-Stokes Informed
792 Deep Learning Framework for Assimilating Flow Visualization Data, *arXiv* (2018).
793 URL <http://arxiv.org/abs/1808.04327>
- 794 [16] A. D. Beck, D. G. Flad, C.-D. Munz, Deep Neural Networks for Data-Driven Turbulence
795 Models, *arXiv* (2018).
796 URL <http://arxiv.org/abs/1806.04482>
- 797 [17] S. L. Brunton, B. Noack, P. Koumoutsakos, Machine learning for fluid mechanics, *Annu. Rev.
798 Fluid Mech.* 52 (2020) 477–508.
- 799 [18] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert,
800 L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driess-
801 che, T. Graepel, D. Hassabis, Mastering the game of go without human knowledge, *Nature*
802 550 (7676) (2017) 354–359.
- 803 [19] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, M. Hutter, Learning
804 agile and dynamic motor skills for legged robots, *Science Robotics* 4 (26) (2019) eaau5872.
- 805 [20] A. Bernstein, E. Burnaev, Reinforcement learning in computer vision, *Proc. SPIE 10696, 10th
806 International Conference on Machine Vision (ICMV 2017)* (2018).
- 807 [21] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra,
808 Continuous control with deep reinforcement learning, *arXiv e-prints* (2015).
809 URL <http://arxiv.org/abs/1509.02971>
- 810 [22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal Policy Optimization
811 Algorithms, *arXiv e-prints* (Jul. 2017). [arXiv:1707.06347](https://arxiv.org/abs/1707.06347).
- 812 [23] P. Ma, Y. Tian, Z. Pan, B. Ren, D. Manocha, Fluid directed rigid body control using deep
813 reinforcement learning, *ACM Transactions on Graphics (TOG)* 37 (4) (2018) 1–11.
- 814 [24] J. Rabault, M. Kuchta, A. Jensen, U. Réglade, N. Cerardi, Artificial neural networks trained
815 through deep reinforcement learning discover control strategies for active flow control, *Journal
816 of Fluid Mechanics* 865 (2019) 281–302.
- 817 [25] H. Tang, J. Rabault, A. Kuhnle, Y. Wang, T. Wang, Robust active flow control over a range
818 of Reynolds numbers using an artificial neural network trained through deep reinforcement
819 learning, *Phys. Fluids* 32 (2020) 053605.
- 820 [26] J. Viquerat, J. Rabault, A. Kuhnle, H. Ghraieb, E. Hachem, Direct shape optimization through
821 deep reinforcement learning, *arXiv preprint arXiv:1908.09885* (2019).

- 822 [27] G. Beintema, A. Corbetta, L. Biferale, F. Toschi, Controlling Rayleigh-Bénard convection
823 via reinforcement learning, arXiv preprint arXiv:2003.14358 (2020).
- 824 [28] H. Kazmi, F. Mehmood, S. Lodeweyckx, J. Driesen, Gigawatt-hour scale savings on a budget
825 of zero: Deep reinforcement learning based optimal control of hot water systems, *Energy* 144
826 (2018) 159–168.
- 827 [29] T. Zhang, J. Luo, P. Chen, J. Liu, Flow rate control in smart district heating systems using
828 deep reinforcement learning, arXiv preprint arXiv:1912.05313 (2019).
- 829 [30] H. Ghraieb, P. Meliga, J. Viquerat, E. Hachem, Optimization and passive flow control using
830 single-step deep reinforcement learning, *Phys. Rev. Fluids* (submitted) (2020).
- 831 [31] C. Gruau, T. Coupez, 3d tetrahedral, unstructured and anisotropic mesh generation with
832 adaptation to natural and multidomain metric, *Comput. Methods Appl. Mech. Engrg.* 194
833 (2005) 4951–4976.
- 834 [32] M. Bernacki, Y. Chastel, T. Coupez, R. Logé, Level set framework for the numerical modelling
835 of primary recrystallization in polycrystalline materials, *Scr. Mater.* 58 (2008) 1129–1132.
- 836 [33] Y. Mesri, H. Dignonnet, T. Coupez, Advanced parallel computing in material forming with
837 CIMLib, *Eur. J. Comput. Mech.* 18 (2009) 669–694.
- 838 [34] T. Coupez, Metric construction by length distribution tensor and edge based error for
839 anisotropic adaptive meshing, *J. Comput. Phys.* 230 (2011) 2391–2405.
- 840 [35] S. Patankar, *Numerical heat transfer and fluid flow*, Taylor and Francis, 1980.
- 841 [36] T. J. R. Hughes, G. R. Feijóo, L. Mazzei, J.-B. Quinicy, The variational multiscale method -
842 a paradigm for computational mechanics, *Comput. Methods Appl. Mech. Engrg.* 166 (1998)
843 3–24.
- 844 [37] R. Codina, Stabilization of incompressibility and convection through orthogonal sub-scales in
845 finite element methods, *Comput. Methods Appl. Mech. Engrg.* 190 (2000) 1579–1599.
- 846 [38] Y. Bazilevs, V. M. Calo, J. A. Cottrell, T. J. R. Hughes, A. Reali, G. Scovazzi, Variational
847 multiscale residual-based turbulence modeling for large eddy simulation of incompressible
848 flows, *Comput. Methods Appl. Mech. Engrg.* 197 (2007) 173–201.
- 849 [39] E. Hachem, B. Rivaux, T. Kloczko, H. Dignonnet, T. Coupez, Stabilized finite element method
850 for incompressible flows with high Reynolds number, *J. Comput. Phys.* 229 (23) (2010) 8643–
851 8665.
- 852 [40] E. Hachem, T. Kloczko, H. Dignonnet, T. Coupez, Stabilized finite element solution to handle
853 complex heat and fluid flows in industrial furnaces using the immersed volume method, *Int.*
854 *J. Numer. Methods Fluids* 68 (2012) 99–121.
- 855 [41] R. Codina, Stabilized finite element approximation of transient incompressible flows using
856 orthogonal subscales, *Comput. Methods Appl. Mech. Engrg.* 191 (2002) 4295–4321.
- 857 [42] E. Hachem, S. Feghali, R. Codina, T. Coupez, Immersed stress method for fluid-structure
858 interaction using anisotropic mesh adaptation, *Int. J. Numer. Meth. Eng.* 94 (2013) 805–825.
- 859 [43] R. Codina, Comparison of some finite element methods for solving the diffusion-convection-
860 reaction equation, *Comput. Methods Appl. Mech. Engrg.* 156 (1998) 185–210.
- 861 [44] S. Badia, R. Codina, Analysis of a stabilized finite element approximation of the transient
862 convection-diffusion equation using an ALE framework, *SIAM Journal on Numerical Analysis*
863 44 (2006) 2159–2197.
- 864 [45] A. N. Brooks, T. Hughes, Streamline upwind/Petrov-Galerkin formulations for convection
865 dominated flows with particular emphasis on the incompressible Navier-Stokes equations,
866 *Comput. Methods Appl. Mech. Engrg.* 32 (1982) 199–259.

- 867 [46] A. Galeão, E. Do Carmo, A consistent approximate upwind Petrov-Galerkin method for
868 convection-dominated problems, *Comput. Methods Appl. Mech. Engrg.* 68 (1988) 83–95.
- 869 [47] E. Hachem, H. Dignonnet, E. Massoni, T. Coupez, Immersed volume method for solving natural
870 convection, conduction and radiation of a hat-shaped disk inside a 3d enclosure, *International*
871 *Journal of numerical methods for heat & fluid flow* (2012).
- 872 [48] E. Hachem, G. Jannoun, J. Veysset, M. Henri, R. Pierrot, I. Poitroult, E. Massoni, T. Coupez,
873 Modeling of heat transfer and turbulent flows inside industrial furnaces, *Simul. Model. Pract.*
874 *Th.* 30 (2013) 35–53.
- 875 [49] I. Goodfellow, Y. Bengio, A. Courville, *The Deep Learning Book*, MIT Press, 2017.
- 876 [50] R. S. Sutton, A. G. Barto, *Reinforcement learning: An introduction*, MIT press, 2018.
- 877 [51] A. Kakade, A natural policy gradient, *Adv. Neural Inf. Process Syst.* 14 (2001) 1531–1538.
- 878 [52] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, P. Abbeel, Trust Region Policy Optimization,
879 arXiv e-prints (Feb. 2015). [arXiv:1502.05477](https://arxiv.org/abs/1502.05477).
- 880 [53] Y. Wang, H. He, X. Tan, Y. Gan, Trust region-guided proximal policy optimization, arXiv
881 preprint arXiv:1901.10314 (2019).
- 882 [54] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse,
883 O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, Y. Wu, Stable baselines,
884 <https://github.com/hill-a/stable-baselines> (2018).
- 885 [55] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, W. Zaremba,
886 Openai gym (2016). [arXiv:arXiv:1606.01540](https://arxiv.org/abs/1606.01540).
- 887 [56] G. de Vahl Davis, I. Jones, Natural convection in a square cavity: a comparison exercise, *Int.*
888 *J. Numer. Methods Fluids* 3 (1983) 227–248.
- 889 [57] H. Dixit, V. Babu, Simulation of high Rayleigh number natural convection in a square cavity
890 using the lattice Boltzmann method, *Int. J. Heat Mass Transfer* 49 (2006) 727–739.
- 891 [58] N. Markatos, K. Pericleous, Laminar and turbulent natural convection in an enclosed cavity,
892 *Int. J. Heat Mass Transfer* 27 (1984) 772–775.
- 893 [59] G. Barakos, E. Mitsoulis, Natural convection flow in a square cavity revisited: laminar and
894 turbulent models with wall functions, *Int. J. Numer. Methods Fluids* 18 (1994) 695–719.
- 895 [60] K. Khanafer, K. Vafai, M. Lightstone, Buoyancy-driven heat transfer enhancement in a two-
896 dimensional enclosure utilizing nanofluids, *Int. J. Heat Mass Transfer* 46 (2003) 3639–3653.
- 897 [61] A. Lazaric, M. Restelli, A. Bonarini, Reinforcement learning in continuous action spaces
898 through sequential Monte Carlo methods, in: *Procs. of the 35th International Conference*
899 *on Machine Learning*, 2018, pp. 4587–4596.
- 900 [62] K. Lee, S. Kim, J. Choi, Deep reinforcement learning in continuous action spaces: a case study
901 in the game of simulated curling, in: *Procs. of the 35th International Conference*
902 *on Machine Learning*, 2018, pp. 4587–4596.
- 903 [63] J. Sari, F. Cremonesi, M. Khalloufi, F. Cauneau, P. Meliga, Y. Mesri, E. Hachem, Anisotropic
904 adaptive stabilized finite element solver for rans models, *Int. J. Numer. Methods Fluids* 86
905 (2018) 717–736.
- 906 [64] G. Guiza, A. Larcher, A. Goetz, L. Billon, P. Meliga, E. Hachem, Anisotropic boundary layer
907 mesh generation for reliable 3D unsteady RANS simulations, *Finite Elem. Anal. Des.* 170
908 (2020) 103345.
- 909 [65] P. Meliga, E. Hachem, Time-accurate calculation and bifurcation analysis of the incompressible
910 flow over a square cavity using variational multiscale modeling, *J. Comput. Phys.* 376 (2019)
911 952–972.