Automating Turbulence Modeling by Multi-Agent Reinforcement Learning

Guido Novati[a], **Hugues Lascombes de Laroussilhe**[a], and **Petros Koumoutsakos**[a,1]

[a]Computational Science and Engineering Laboratory, Clausiusstrasse 33, ETH Zürich, CH-8092, Switzerland

This manuscript was compiled on October 26, 2020

Turbulent flow models are critical for applications such as aircraft design, weather forecasting and climate prediction. Existing models are largely based on physical insight and engineering intuition. More recently, machine learning has been contributing to this endeavor with promising results. However all efforts have focused on supervised learning, which encounters difficulties to generalize beyond the training data.
Here, we introduce multi-agent reinforcement learning (MARL) as an automated discovery tool of turbulence models. We demonstrate the potential of this approach on Large Eddy Simulations of isotropic turbulence, using as reward the recovery of the statistical properties of Direct Numerical Simulations. The closure model is a control policy enacted by cooperating agents, which detect critical spatio-temporal patterns in the flow field to estimate the unresolved sub-grid scale physics. Results obtained with MARL algorithms based on experience replay compare favorably with established modeling approaches. Moreover, we show that the learned turbulence models generalize across grid sizes and flow conditions.

Multi-agent Reinforcement Learning | Turbulence Modeling | Large-eddy Simulations

The prediction of the statistical properties of turbulent flows is critical for engineering (cars to nuclear reactors), science (ocean dynamics to astrophysics) and government policy (climate and weather forecasting). Over the last sixty years we have increasingly relied for such predictions on simulations based on the numerical integration of the Navier-Stokes equations. Today we can perform simulations using trillions of computational elements and resolve flow phenomena at unprecedented detail. However, despite the ever increasing availability of computing resources, most simulations of turbulent flows require the adoption of models to account for the spatio-temporal scales that cannot be resolved. Over the last few decades, the development of turbulence models has been the subject of intense investigations that have relied on physical insight and engineering intuition. Recent advances in machine learning and in the availability of data have offered new perspectives (and hope) in developing data-driven turbulence models. The study of turbulent flows is rooted in the seminal works of Kolmogorov on statistical analysis [1]. These flows are characterized by vortical structures, and their interactions, exhibiting a broad spectrum of spatio-temporal scales [2, 3]. At one end of the spectrum we encounter the integral scales, which depend on the specific forcing, flow geometry, or boundary conditions. At the other end are the Kolmogorov scales at which turbulent kinetic energy is dissipated. The handling of these turbulent scales provides a classification of turbulence simulations: Direct Numerical Simulations (DNS), which use a sufficient number of computational elements to represent all scales of the flow field, and simulations using turbulence models

where the equations are solved in relatively few computational elements and the non-resolved terms are described by closure models. While the flow structures at Kolmogorov scales are statistically homogeneous and dissipate energy, most of the computational effort of DNS [4] is spent in attempting to fully resolve them. DNS [5] have provided us with unique insight into the physics of turbulence that can lead in turn to effective turbulence modeling. However, it is well understood that for the foreseeable future DNS will not be feasible at resolutions necessary for engineering applications. In the development of turbulence models [6] two techniques have been dominant: Reynolds Averaged Navier-Stokes and Large-eddy Simulations (LES) [7] in which only the large scale unsteady physics are explicitly computed whereas the sub grid-scale (SGS), unresolved, physics are modeled. In LES, classic approaches to the explicit modeling of SGS stresses include the standard [8] and the dynamic Smagorinsky model [9, 10]. In the last 50 years SGS models have been constructed using physical insight, numerical approximations and often problem-specific intuition. The first efforts to develop models for turbulent flows using machine learning [11, 12] were hindered by the available computing power and the convergence of the training algorithms. Recent advances in hardware and algorithms have fueled a broad interest in the development of data-driven turbulence models [13].

To date, to the best of our knowledge, all data-driven turbulence closure models are based on supervised learning. In LES, early approaches [14] trained a NN to emulate and speed-up a conventional, but computationally expensive, SGS model. More recently, data-driven SGS models have been trained by supervised learning to predict the "perfect" SGS terms computed from filtered DNS data [15, 16]. Variants include deriving the target SGS term from optimal estimator theory [17] and reconstructing the SGS velocity field as a deconvolution operation, or inverse filtering [18, 19]. In supervised learning, the parameters of the NN are commonly derived by minimizing the model prediction error via a gradient descent algorithm. As the error is required to be differentiable with respect to the model parameters, and due to the computational challenge of obtaining chain-derivatives through a flow solver [20], supervised learning approaches often rely on one-step target values for the model (e.g. SGS stresses computed from filtered DNS). Such *a-priori* testing measures the accuracy of the derived model in predicting the target values from a database of reference simulations, typically obtained via DNS. After training, *a-posteriori* testing is performed by integrating in time the flow equations along with the learned closure and comparing the obtained statistical quantities to those from DNS or other

references. We remark that in the case of a single-step cost function, the resultant NN model is not trained to compensate for the evolution of discrepancies between DNS and LES data and the compounding errors. This critical issue of SGS models derived by supervised learning has been exposed by studies that perform *a posteriori* testing (21). *A-priori* perfect SGS models may be structurally unstable, accumulate high-spatial frequency errors and diverge from the original trajectory under small perturbations (22, 23). Moreover, models trained to reproduce a particular quantity of interest may worsen the accuracy of other physical quantities (15).

In this work, we address these challenges by introducing Reinforcement Learning (RL) as a framework for the automated discovery of closure models for non-linear conservation laws. Specifically, we analyze the potential of RL to control under-resolved simulations (LES) of isotropic turbulence by locally adapting the coefficients of the eddy viscosity closure model, with the objective of accurately reproducing the energy spectrum predicted by DNS. Two characteristics of RL make it particularly suited to the task. First, RL casts the closure problem in terms of the actions of an agent that learns to optimize their long-term consequences on the environment. In RL, training is not performed on a database of reference data, but by integrating in time the parametric model. Consequently, the RL framework overcomes the above mentioned distinction between *a priori* and *a posteriori* evaluation and accounts for compounding modeling errors. Moreover, the performance of a RL strategy is not measured by a differentiable objective function but by a cumulative reward. Supervised learning approaches that train a model to recover SGS quantities computed from filtered DNS simulations necessitate the computational capabilities of fully resolving the flow simulation. This is not required in RL, as the reward can be a measure of the similarity between the statistics of a quantity of interest produced by the model and reference data, which may even be obtained from experiments. Finally, we remark that the proposed RL framework is not restricted to simulations of the Navier-Stokes equations and is readily adaptable to other non-linear conservation laws.

Multi-agent RL for sub-grid scale modeling

RL is a computational framework for control problems (24) which implies goal-directed interactions of an agent with its environment. RL is at the core of some of the seminal results of machine learning, in applications including games (25, 26) and robotics (27, 28). In RL the agent performs *actions* that affect its environment. The agent's actions are contingent on its *state* and the performance is measured via scalar *reward* functions. By acquiring experience, the agent learns a policy ($\pi(a|s)$) from which it samples actions that maximize the long term, cumulative rewards. In recent years, RL has been making inroads in the field of flow control (29). By interacting with the flow field, agents trained through RL were able to gather relevant information and optimize their decision process to perform collective swimming (30), soar (31), minimize their drag (32, 33), delay the onset of instabilities (34), or reach a target location (35, 36).

A key aspect of RL is the representation of the policy function. Deep RL algorithms train NN to represent the policy ($\pi_w(\cdot|\mathbf{s})$, with parameters w), bypassing the need for tabular or expertly designed state representations (37). In the present study, a policy-network is used to sample the dissipation coefficient C_s^2 of the Smagorinsky SGS model. We emphasize that the interface of RL with the flow solver has considerable effect on the computational efficiency of the resulting model. As an example, following the common practice in video games (25), the state $\mathbf{s}$ of the agent could be defined as the full three-dimensional flow field at a given time-step and the action as the SGS closure for all grid-points. With such an architecture, the dimensionality of both state and action spaces would scale with the number of degrees of freedom of the simulation. As a consequence, the closure model would be mesh-size dependent, would involve training a very large NN, and the memory needed to store the experiences of the agent would be prohibitively large. Here, we overcome these issues through multi-agent reinforcement learning (MARL).

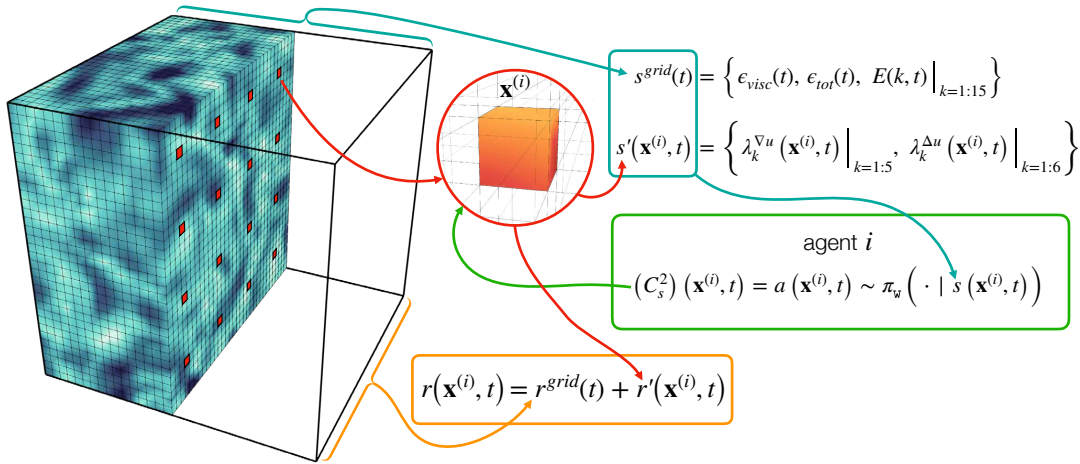In MARL, the N_{agents} agents are dispersed in the simula-

Fig. 1. Schematic of the integration of MARL with the flow solver. The agents (located at the red blocks) compute the SGS dissipation coefficient C_s^2 for each grid-point of the simulation by sampling a shared control policy $\pi_w(\cdot|\mathbf{s}(\mathbf{x},t))$. The state $\mathbf{s}(\mathbf{x},t)$ of agent i at time t is defined by both local variables (i.e. $\mathbf{s}'(\mathbf{x},t)$, which includes the invariants of the gradient $\lambda_k^{\nabla\mathbf{u}}$ and Hessian $\lambda_k^{\Delta\mathbf{u}}$ of the velocity field computed at the agents' location $\mathbf{x}^{(i)}$), and global variables (i.e. $\mathbf{s}^{grid}(t)$, composed by the energy spectrum up to the Nyquist frequency, the rate viscous dissipation ϵ_{visc} and the total dissipation rate ϵ_{tot}). Depending on the accuracy of the resulting LES simulation, the agents receive a scalar reward $r(\mathbf{x},t)$ which can be similarly defined from both local and global components.

tion domain (Fig. 1). Each agent (i, with spatial coordinate $\boldsymbol{x}^{(i)}$) performs a localized action based on information about the state of the flow field $s(\boldsymbol{x}^{(i)}, t) \in \mathbb{R}^{dim_S}$, which is encoded by a small set of local and global variables. We embed tensorial invariance into the NN inputs (38) by selecting as local variables of the state vector the 5 invariants (39) of the gradient ($\lambda_k^{\nabla \mathbf{u}}$) and the 6 invariants of the Hessian of the velocity field ($\lambda_k^{\Delta \mathbf{u}}$). These are computed at the agents' location and non-dimensionalized with K/ϵ. The global components of the state are the modes of the energy spectrum up to the training grid's Nyquist frequency $N_{nyquist}$ (non-dimensionalized with u_η), the rate of viscous dissipation (ϵ_{visc}/ϵ) and the total dissipation (ϵ_{tot}/ϵ) relative to the turbulent energy injection rate. The training phase is performed on a Cartesian mesh of size $N = 32^3$, with a pressure-projection scheme, second-order discretization of the spatial derivatives, and second-order explicit Runge-Kutta time integration. Because $N_{Nyquist} = 15$, we have state dimensionality $dim_S = 28$. These are far fewer variables than would be required by encoding as a state the entire velocity field of the flow ($dim_S = 3 \cdot 32^3$).

MARL advances in turns by updating the dissipation coefficients $C_s^2(\boldsymbol{x}, t)$ for the whole flow and integrating the LES in time for Δt_{RL}, until $t = T_{end}$ or any numerical instability arises. At the start of every turn, each agent i measures its state and selects an action $a(\boldsymbol{x}^{(i)}, t) \in \mathbb{R}$ by sampling a Gaussian policy: $a(\boldsymbol{x}^{(i)}, t) \sim \pi_{\mathtt{w}}(\cdot \mid s(\boldsymbol{x}^{(i)}, t)) \equiv \mathcal{N}[\mu_{\mathtt{w}}\left(s(\boldsymbol{x}^{(i)}, t)\right), \sigma_{\mathtt{w}}^2\left(s(\boldsymbol{x}^{(i)}, t)\right)]$. The actions are used to compute $C_s^2(\boldsymbol{x}, t)$ for each grid point by linear interpolation:

$$C_s^2(\boldsymbol{x}, t) = \sum_{i=1}^{N_{agents}} a(\boldsymbol{x}^{(i)}, t) \prod_{j=1}^{3} \max \left\{ 1 - \frac{|x_j - x_j^{(i)}|}{\Delta_{agents}}, \ 0 \right\}, \quad [1]$$

where $x_j^{(i)}$ is the j-th Cartesian component of the position vector of agent i, and $\Delta_{agents} = 2\pi / \sqrt[3]{N_{agents}}$ is the distance between agents. If $N_{agents} = N$, no interpolation is required.

Increasing N_{agents} improves the adaptability of MARL to localized flow features. However, the actions of other agents are confounding factors that may increases the update variance (40). For example, if the C_s^2 coefficient selected by one agent causes numerical instability, all agents would receive negative feedback, regardless of their choices. These challenges are addressed by performing policy optimization with the Remember and Forget Experience Replay algorithm (ReF-ER) (41). Three features of ReF-ER make it particularly suitable for MARL and the present task: First, as it relies on Experience Replay (ER) (42), it reuses experiences over multiple policy iterations and increases the accuracy of gradient updates by computing expectations from uncorrelated experiences. Moreover, ReF-ER is inherently stable and has been shown to reach, and even surpass, the performance of state of the art RL algorithms and optimal control (36) methods on several benchmark problems. Finally, and crucially for MARL, ReF-ER explicitly controls the pace of policy changes. We found that ReF-ER, with strict constraints on the policy updates from individual experiences, is necessary to stabilize training and compensate for the imprecision of the single-agent update rules. For a thorough description of ReF-ER see the Methods and the original paper (41).

MARL finds the parameters $\mathtt{w}$ that maximize the expected sum of rewards over each simulation $J(\mathtt{w}) = \mathbb{E}_{\pi_{\mathtt{w}}} \left[\sum_{t=1}^{T_{end}} r_t \right]$.

In the present study, the parameters $\mathtt{w}$ are shared by all agents. Their aggregate experiences are collected in a shared dataset and used to computes updates according to ReF-ER (Fig. 5). We define reward functions with the objective of obtaining policies $\pi_{\mathtt{w}}$ that yield stable LES and exhibit statistical properties that closely match those of DNS. We find that the distribution of energy spectrum $E(k)$ of isotropic turbulence computed by DNS is well approximated by a log-normal distribution (see Methods and Fig. 6) such that $\log E_{DNS}^{Re_\lambda} \sim \mathcal{N}\left(\mu_{DNS}^{Re_\lambda}, \Sigma_{DNS}^{Re_\lambda}\right)$, where $\mu_{DNS}^{Re_\lambda}$ is the average log-energy spectrum for a given Re_λ and $\Sigma_{DNS}^{Re_\lambda}$ is its covariance matrix. When comparing SGS models and formulating objective functions, we rely on a regularized log-likelihood:

$$\widetilde{LL}(E_{LES}^{Re_\lambda} | E_{DNS}^{Re_\lambda}) = \log \mathcal{P}(E_{LES}^{Re_\lambda} | E_{DNS}^{Re_\lambda}) / N_{nyquist}. \quad [2]$$

Here the probability metric is

$$\mathcal{P}(E_{LES}^{Re_\lambda} | E_{DNS}^{Re_\lambda}) \propto \exp \left[-\frac{1}{2} \left(\log E_{LES}^{Re_\lambda} - \bar{\mu}_{DNS}^{Re_\lambda} \right)^T \right.$$
$$\left. \left(\bar{\Sigma}_{DNS}^{Re_\lambda} \right)^{-1} \left(\log E_{LES}^{Re_\lambda} - \bar{\mu}_{DNS}^{Re_\lambda} \right) \right] \quad [3]$$

with E_{LES} the LES energy spectrum, $\bar{\mu}_{DNS}^{Re_\lambda}$ and $\bar{\Sigma}_{DNS}^{Re_\lambda}$ the target statistics up to $N_{nyquist}$.

We consider two SGS models derived from MARL, each corresponding to a reward function of the form $r(\boldsymbol{x}^{(i)}, t)$. The first ($\pi_{\mathtt{w}}^G$) is defined by rewards $r^G(\boldsymbol{x}^{(i)}, t)$ based on the error in the Germano identity (9) which states that the sum of resolved and modeled contributions to the SGS stress tensor should be independent of LES resolution. The second ($\pi_{\mathtt{w}}^{LL}$) is defined by rewards $r^{LL}(\boldsymbol{x}^{(i)}, t)$ which strongly penalize discrepancies from the target energy spectra. While r^G is computed locally for each agent, r^{LL} equal for all agents. We remark that the target statistics involve spatial and temporal averages and can be computed from a limited number of DNS, which for this study are four orders of magnitude more computationally expensive than LES.

LES modeling of DNS

The Taylor-Reynolds number (Re_λ) characterizes the breadth of the spectrum of vortical structures present in a isotropic turbulent flow (2, 3). Figure 2 illustrates the challenge in developing a reliable SGS model for a wide range of Re_λ and for a severely under-resolved grid. Only the large eddies are resolved and for the lower Reynolds numbers (e.g. $Re_\lambda = 65$) the SGS model is barely able to represent the flow features of DNS. For Reynolds numbers beyond $Re_\lambda = 111$ the fluctuating vortical structures that characterize the turbulent flow field occur at length-scales that are much smaller than the LES grid-size. As a consequence, an increasing portion of energy dissipation is due to SGS effects, which leads to instability if these are not accurately modeled.

A natural way to assess whether the turbulence model accurately reproduces the energy transfer to SGS motions is through the energy spectra. In Fig. 3a, we compare the time-averaged spectra obtained by DNS to those obtained by LES with several SGS closure models. More specifically, we show the first $N_{nyquist}$ modes of the energy spectra, and their
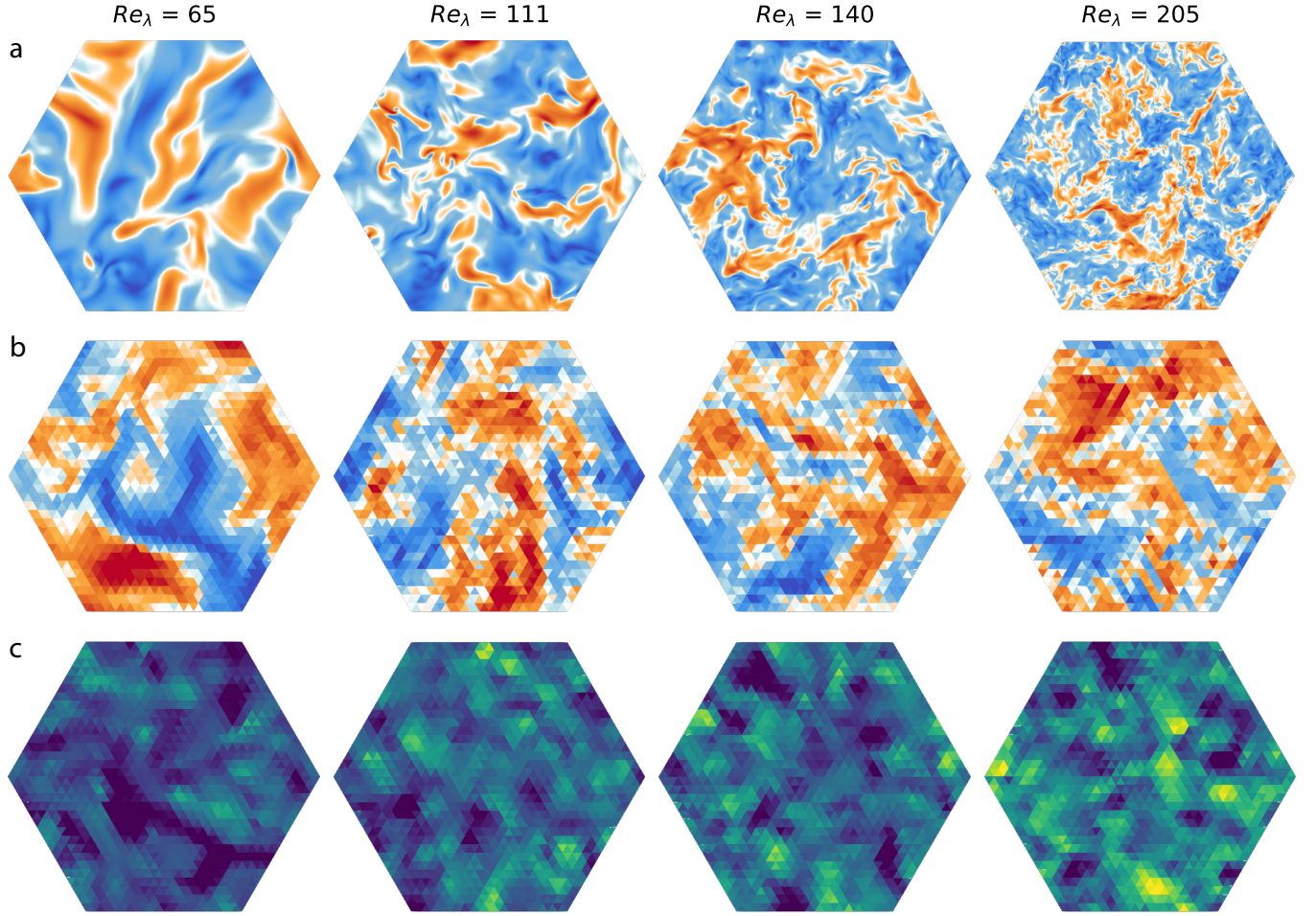
$Re_\lambda = 65$　　$Re_\lambda = 111$　　$Re_\lambda = 140$　　$Re_\lambda = 205$

Fig. 2. Visualizations of simulations of isotropic turbulence. a, Representative contours of momentum flux across a diagonal slice ($x + y + z = 0$) of the cubical domain ($\boldsymbol{u} \cdot \boldsymbol{n}$, blue and orange indicate negative and positive fluxes) for DNS of isotropic turbulence with resolution 1024^3, **b**, for LES with resolution 32^3 and SGS modeling with a RL policy trained for r^{LL}. **c**, Contours of the Smagorinsky coefficient C_s^2 across the same diagonal slice of the LES (blue and yellow indicate low and high values respectively).

normalization with the mean and standard deviation of the energy computed through DNS (Fig. 3**b**). This measure quantifies the contributions of individual modes to the objective log-likelihood (Eq. 2). A perfect SGS model would produce a spectrum with time-averaged $E_{LES}^{Re_\lambda}(k)$ with the same statistics as that from a DNS. We consider the two classical approaches, the standard Smagorinsky model (SSM-with an empirically tuned constant dissipation coefficient C_s), and the dynamic Smagorinsky model (DSM- with an adaptive coefficient derived from the Germano identity). The two models serve as a reference for the accuracy of SGS models derived through MARL, identified by the policies $\pi_{\mathbf{w}}^{LL}$ and $\pi_{\mathbf{w}}^{G}$. We remark that the models are evaluated in flows with Reynolds numbers which were not presented during training. The amount of SGS dissipation, as well as the the numerical scales of the flow quantities, and of the RL state components, vary with Re_λ. Therefore, the results for $Re_\lambda = 82$, 111, and 151 measure the MARL model accuracy for dynamical scales that are interposed with the training ones, while the results for $Re_\lambda = 60$, 190 and 205 measures the ability of the MARL models to generalize beyond the training experiences.

The MARL model trained to satisfy the Germano identity highlights an important consequence of RL maximizing long term rewards. DSM, which minimizes the instantaneous Germano-error, exhibits growing energy build-up at high-frequencies, which causes numerical instability at higher Re_λ. In fact, the Germano identity is not expected to be accurate for severely under-resolved LES. Conversely, $\pi_{\mathbf{w}}^{G}$, which minimizes the error over all future steps, over-estimates the dissipation coefficient, smoothing the velocity field, and making it easier for future actions to satisfy the Germano identity. This can be otherwise observed by Fig. 3**c**, which shows the empirical distribution of Smagorinsky coefficients chosen by the SGS models. While outwardly DSM and $\pi_{\mathbf{w}}^{G}$ minimize the same relation, $\pi_{\mathbf{w}}^{G}$ introduces much more artificial viscosity.

The policy $\pi_{\mathbf{w}}^{LL}$, which directly maximizes the similarity between quantity of interest (i.e. energy spectra) obtained by MARL and those of DNS, produces the SGS model of the highest quality. While its accuracy is similar to that of DSM for lower values of Re_λ, $\pi_{\mathbf{w}}^{LL}$ avoids energy build-up and remains stable up to $Re_\lambda = 205$, well beyond the maximum training $Re_\lambda = 163$. Higher Reynolds numbers were not tested as they would have required increased spatial and temporal resolution to carry out accurate DNS, with prohibitive computational cost. We evaluate the difficulty of generalizing beyond the training data by comparing $\pi_{\mathbf{w}}^{LL}$ to a policy fitted exclusively
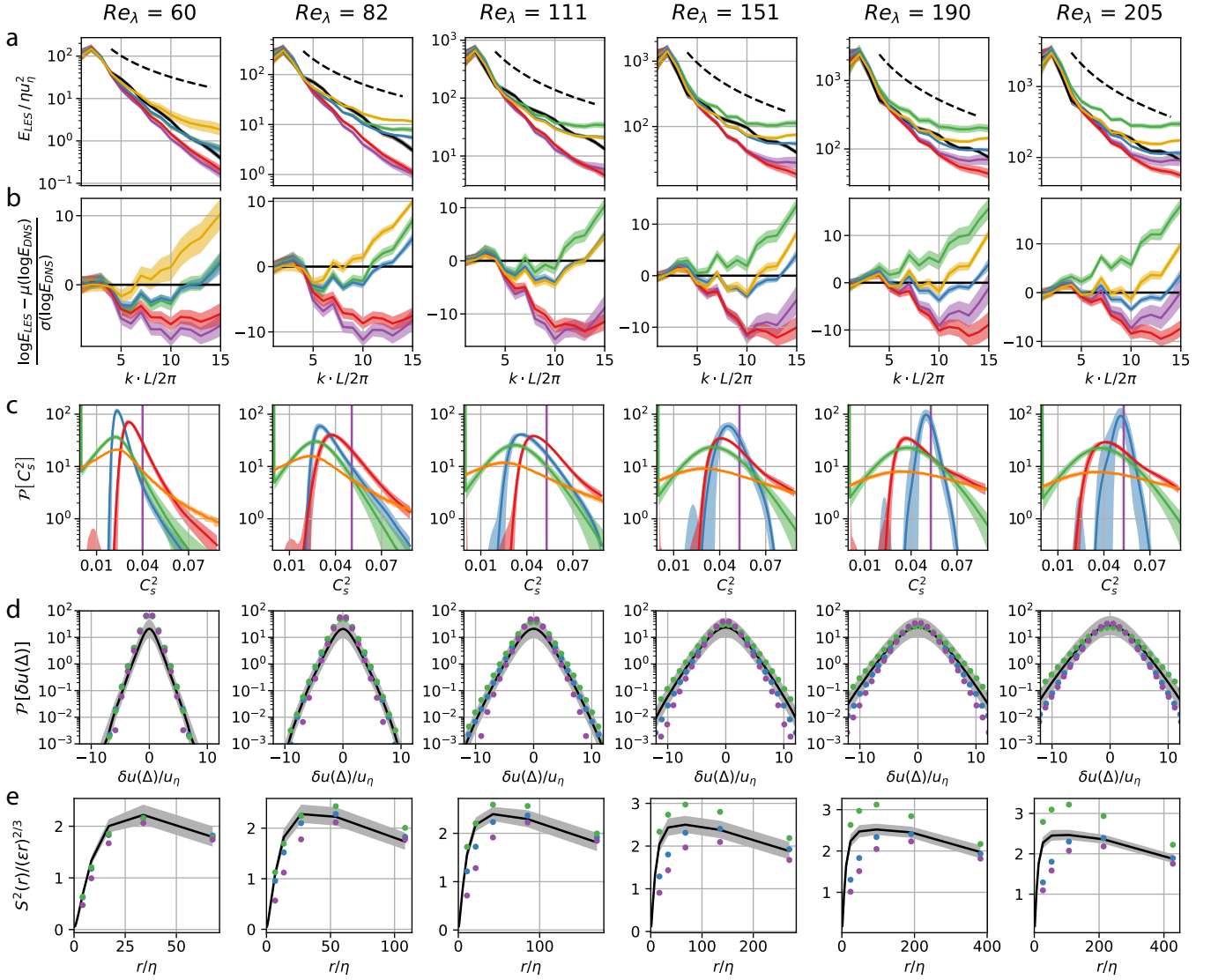
Fig. 3. Comparison of SGS models for values of Re_λ that were *not* included during the MARL training. a, Energy spectra for (——) DNS, (——) SSM, (——) DSM, (——) MARL policy $\pi_{\mathbf{w}}^{LL}$, (——) MARL policy $\pi_{\mathbf{w}}^{G}$, and (——) MARL policy $\pi_{\mathbf{w}}^{LL,111}$ trained exclusively from data for $Re_\lambda = 111$, compared to (– –) the -5/3 Kolmogorov scaling. **b**, Log-energy spectra normalized by the DNS mean and the standard deviation. **c**, Empirical probability distributions of the Smagorinsky model coefficient C_s^2 for the first four models (same colors) compared to (——) the SGS dissipation coefficient computed from DNS filtered to LES resolution. **d**, distribution of longitudinal velocity increments $\delta u(r) = [\mathbf{u}(\mathbf{x} + \mathbf{r}) - \mathbf{u}(\mathbf{x})] \cdot \hat{\mathbf{r}}$ for r equal to the LES grid size Δ and, **e**, second-order velocity structure function $S^2(r) = \langle \delta u(r)^2 \rangle$ for (•) $\pi_{\mathbf{w}}^{LL}$, (•) DSM, and (•) SSM compared to DNS. In all cases, lines represent averages and contours represent intervals of one standard deviation.

for $Re_\lambda = 111$ ($\pi_{\mathbf{w}}^{LL,\ 111}$). Fig. 3**b** shows the specialized policy to have comparable accuracy at $Re_\lambda = 111$, but becomes rapidly invalid when varying the dynamical scales. This result supports that data-driven SGS models should be trained on varied flow conditions rather than with a training set produced by a single simulation.

From Fig. 3**c**, we observe that $\pi_{\mathbf{w}}^{LL}$ achieves its accuracy by producing a narrower distribution of C_s^2. In this respect, $\pi_{\mathbf{w}}^{LL}$ stands in contrast to a model trained by supervised learning to reproduce the SGS stresses computed from filtered DNS. By filtering the DNS results to the same resolution as the LES, thus isolating the unresolved scales, we emulate the distribution of C_s^2 that would be produced by a SGS model trained by supervised learning. We find that such model would have lower SGS dissipation than both DSM and $\pi_{\mathbf{w}}^{LL}$, suggesting that, with the present numerical discretization

schemes, it would produce numerically unstable LES. In fact, it is well known that the numerical discretization errors due to the coarse LES resolution, which may not be anticipated when using filtered DNS as training data, may be comparable or larger than the modeling errors (i.e. the errors due to the SGS model) [43]. This further highlights the unique ability of MARL to systematically optimize high-level objectives, such as matching the statistics of DNS, and suggests its potential in deriving data-driven closure equations.

Generalization beyond the training objective

The energy spectrum is just one of many statistical quantities that a physically sound LES should accurately reproduce. In fact, MARL in order to maximize its objective function may inadvertently sacrifice the physical soundness of other
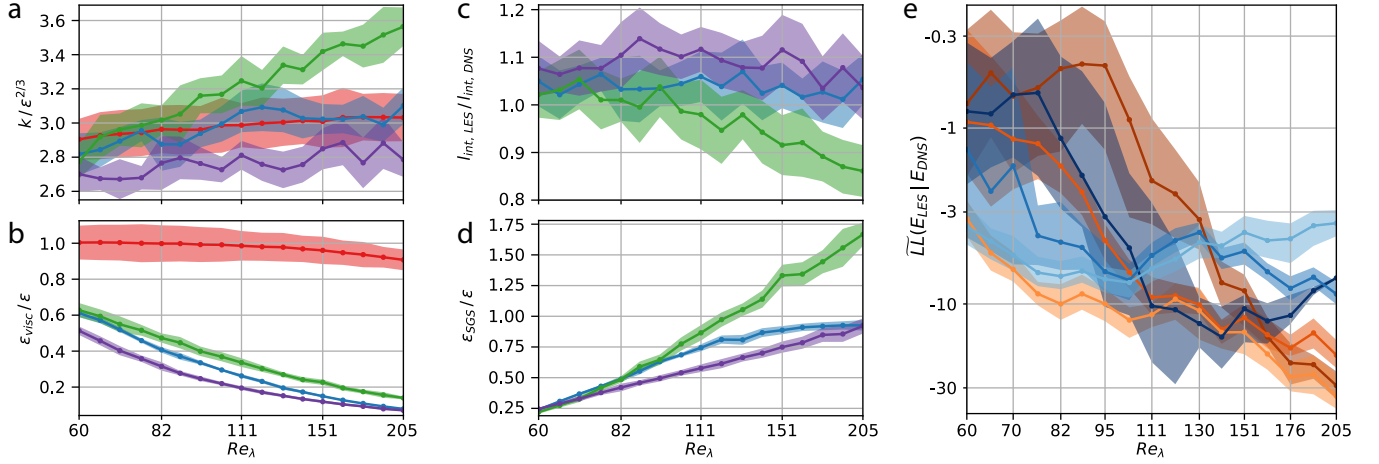
Fig. 4. Measurements of the reliability of the MARL model beyond the training objective. Integral properties of LES: **a**, turbulent kinetic energy, **b**, integral length scale, **c**, ratio of viscous dissipation to energy injection, and **d**, ratio of SGS dissipation to energy injection. The remaining component of energy dissipation is due to numerical discretization. (—•—) SSM, (—•—) DSM, (—•—) MARL policy $\pi_{\mathtt{w}}^{LL}$, and (—•—) DNS simulation when applicable. **e**, Accuracy across grid-resolutions measured as log-likelihood of LES spectra with respect to DNS statistics (Eq. 2) for DSM at resolutions (—•—) $N=32^3$, (—•—) $N=64^3$, (—•—) $N=128^3$ and MARL policy $\pi_{\mathtt{w}}^{LL}$ at resolutions (—•—) $N=32^3$, (—•—) $N=64^3$, (—•—) $N=128^3$. In all cases, data points correspond the temporal averages, and contours denote intervals of one standard deviation.

quantities. Fig. 3**d** shows the distribution of relative velocity between two adjacent points of the LES grid. We see that the tails of the distribution are tapered by $\pi_{\mathtt{w}}^{LL}$ and SSM in order to maintain stability. Fig. 3**e** shows the second-order velocity structure function $S^2(r)$, which is the covariance of the velocity between two points separated by a distance r. According to Kolmogorov's hypothesis, for $r \gg \eta$, the structure function should have scaling behavior $S^2(r) \propto r^{\zeta(p)}$ with coefficient depending only on the energy flux ϵ (1, 3). As expected from DNS, the scaling of $S^2(r)$ approaches a constant value regardless of Re_λ. In Fig. 4**a-d** we compare the total kinetic energy (TKE), the characteristic length scale of the largest eddies (l_{int}), and dissipation rates among LES models and DNS. While in DNS energy is dissipated entirely by viscosity (and if under-resolved by numerical diffusion), in LES the bulk of viscous effects occur at length-scales below the grid size, especially at high Re_λ. We find that for $Re_\lambda = 205$ the stable SGS models dissipate approximately 10 times more energy than viscous dissipation, which underlines the crucial role of turbulence modeling. Up to the point of instability at $Re_\lambda \approx 100$, DSM yields a good estimate for TKE and l_{int}. Beyond that value, the artificial energy created by numerical instabilities causes the SGS dissipation to increase past the energy injection rate ϵ. Despite these quantities not being directly included in the rewards, they are all correctly recovered by the MARL model $\pi_{\mathtt{w}}^{LL}$.

Finally, we evaluate MARL across grid resolutions. Because of the design of the MARL framework, the policy $\pi_{\mathtt{w}}^{LL}$, trained for a single grid-size $N = 32^3$, is valid as long as there exist at least 15 modes of the energy spectrum. In Fig. 4**e** we compare the log-likelihood of DSM and MARL models given the DNS statistics for $N = 32^3$, 64^3 and 128^3. Accordingly, we increase the number of agents per simulation by a factor of 8 and 64 to keep constant the density of agents in the grid. We remark that LES at finer resolutions more accurately represent large-scale statistics, but this is not reflected in the values of the log-likelihood. In fact, at finer resolutions more modes are included and $\widetilde{LL}$ is dominated by errors at the high frequencies. Moreover, because only the first 15

components of the spectrum are available to $\pi_{\mathtt{w}}^{LL}$, MARL agents were not trained to take into account higher energy modes. Finer resolutions are able to capture sharper velocity gradients not experienced during training. As a consequence, $\pi_{\mathtt{w}}^{LL}$ was found to be more diffusive than DSM at the higher frequencies. Nevertheless, the SGS model derived by MARL remains stable throughout the evaluation and markedly more accurate than DSM, especially at higher values of Re_λ.

Discussion

This paper introduces multi-agent RL (MARL) to automate the discovery of closure models in simulations of turbulent flows. We demonstrate the feasibility and potential of this approach on large-eddy simulations (LES) of forced isotropic turbulence. MARL develops the sub-grid scale (SGS) closure as a control policy enacted by cooperating agents. The agents are incorporated into the flow solver, observe local (e.g. invariants of the velocity gradient) as well as global (e.g. the energy spectrum) flow quantities, and accordingly compute SGS residual-stresses through the Smagorinsky (8) formulation. The Remember and Forget Experience Replay (ReF-ER) method is instrumental for the present study by combining the sample-efficiency of ER and the stability of constrained policy updates (41).

We believe that the learning algorithms and results of the present study open new horizons for turbulence modeling efforts. RL maximizes high-level objectives computed from direct application of the learned model and produces SGS models that are stable under perturbation and resistant to compounding errors. Here, MARL minimizes the discrepancies between the energy spectra of LES and that computed from orders of magnitude more computationally expensive, fully resolved simulations (DNS). Access to DNS targets allowed us to vary systematically Re_λ and analyze the accuracy and generality of the trained model with respect to multiple quantities of interest.

New questions emerge from integrating deep learning and

turbulence modeling. In the present study, the control policies trained by MARL (e.g. $\pi_{\mathtt{w}}^{LL}$) are functions with 28-dimensional input and 6'211 parameters which encode the complex correlations between input and eddy-viscosity coefficient. In fact, the present policies may not be readily transferrable to another flow solver which uses a different numerical discretization. While machine learning approaches can be faulted for the lack of generality guarantees and for the difficulty of interpreting the trained model, we envision that sparse RL methods could enable the analysis of causal processes in turbulent energy dissipation and the distillation of mechanistic models. Moreover, the MARL framework, when applied to a new solver, will again learn to compensate for its modeling errors, a capability that may not be readily available to other closure models. Finally, data-driven models, optimized for one type of reward may violate other physical constraints and invariants of the flow. Empirical results indicate that this issue does not affect the present results, but it remains an open question.

At the same time, the MARL framework opens many new directions for building upon the present results. MARL offers new perspectives on addressing classic challenges of LES, such as wall-layer modeling and inflow boundary conditions (44). Moreover, the ability of performing DNS is not required to train RL models. Energy spectra, wall shear stresses or drag coefficients may be measured from experiments and used to train SGS models for LES, avoiding the computational expense of DNS. In fact, many turbulent flows are not stationary, homogeneous or isotropic. It may be prohibitively expensive to measure statistical properties for such flows through multiple realizations of DNS. It is an open, and exciting, research direction to examine whether instantaneous, noisy, experimental measurements of multiple quantities of interest, translated into a reward function, may be used to successfully train SGS closures through RL.

Materials and Methods

The Reinforcement Learning framework RL algorithms advance by trial-and-error exploration and are known to require large quantities of interaction data, in this case acquired by performing thousands of LES with modest but non-negligible cost (which is orders of magnitude higher than the cost of ordinary differential equations or many video games). Therefore, the design of a successful RL approach must take into account the actual computational implementation. Here we rely on the open-source RL library **smarties**, which was designed to ease high-performance interoperability with existing simulation software. **smarties** efficiently leverages the computing resources by separating the task of updating the policy parameters from the task of collecting interaction data (Fig. 5). The flow simulations are distributed across $N_{workers}$ computational nodes ("workers"). The workers collect, for each agent, experiences organized into episodes:

$$\tau_i = \left\{ s_t^{(i)}, r_t^{(i)}, \mu_t^{(i)}, \sigma_t^{(i)}, a_t^{(i)} \right\}_{t=0:T_{end}^{(i)}}$$

where t tracks in-episode RL steps; $\mu_t^{(i)}$ and $\sigma_t^{(i)}$ are the statistics of the Gaussian policy used to sample $a_t^{(i)}$ with the policy parameters available to the worker at time step t of the i-th episode, often termed "behavior policy" $\beta_t^{(i)} \equiv \mathcal{N}(\mu_t^{(i)}, \sigma_t^{(i)})$ in the off-policy RL literature. When a simulation concludes, the worker sends one episode per agent to the central learning process ("master") and receives updated policy parameters. Therefore each simulation produces N_{agents} episodes. The master stores the episodes into a Replay Memory (RM), which is sampled to update the policy parameters according to ReF-ER (41).

ReF-ER can be combined with many ER-based RL algorithms as it consists in a modification of the optimization objective. Here we employ V-RACER, a variant of off-policy policy optimization proposed in conjunction with ReF-ER which supports continuous state and action spaces. Because the cooperating agents do not explicitly coordinate their actions, the algorithm is unchanged from its initial publication. V-RACER trains a Neural Network (NN) which, given input s_t, outputs the mean $\mu_{\mathtt{w}}(s_t)$ and standard deviation $\sigma_{\mathtt{w}}(s_t)$ of the policy $\pi_{\mathtt{w}}$, and a state-value estimate $v_{\mathtt{w}}(s_t)$. One gradient is defined per NN output. The statistics $\mu_{\mathtt{w}}$ and $\sigma_{\mathtt{w}}$ are updated with the *off-policy policy gradient* (*off-PG*) (45):

$$g^{\mathrm{pol}}(\mathtt{w}) = \mathbb{E}\left[\left. (\hat{q}_t - v_{\mathtt{w}}(s_t)) \frac{\pi_{\mathtt{w}}(a_t|s_t)}{\mathcal{P}(a_t|\mu_t, \sigma_t)} \nabla_{\mathtt{w}} \log \pi_{\mathtt{w}}(a_t|s_t) \right| \right.$$
$$\left. \{s_t, r_t, \mu_t, \sigma_t, a_t, \hat{q}_t\} \sim RM \right]. \quad [4]$$

Here $\mathcal{P}(a_t|\mu_t, \sigma_t)$ is the probability of sampling a_t from a Gaussian distribution with statistics μ_t and σ_t, and $\hat{q}_t$ estimates the cumulative rewards by following the current policy from (s_t, a_t) and is computed with the Retrace algorithm (46):

$$\hat{q}_t = r_{t+1} + \gamma v_{\mathtt{w}}(s_{t+1}) + \gamma \min\left\{ 1, \frac{\pi_{\mathtt{w}}(a_t|s_t)}{\mathcal{P}(a_t|\mu_t, \sigma_t)} \right\} [\hat{q}_{t+1} - v_{\mathtt{w}}(s_{t+1})],$$
$$[5]$$

with $\gamma = 0.995$ the discount factor for rewards into the future. Equation 5 is computed via backward recursion when episodes are entered into the RM (note that $\hat{q}_{T_{end}} \equiv 0$), and iteratively updated as individual steps are sampled. Retrace is also used to derive the gradient for the state-value estimate:

$$g^{val}(\mathtt{w}) = \mathbb{E}\left[\left. \min\left\{ 1, \frac{\pi_{\mathtt{w}}(a_t|s_t)}{\mathcal{P}(a_t|\mu_t, \sigma_t)} \right\} (\hat{q}_t - v_{\mathtt{w}}(s_t)) \right| \right.$$
$$\left. \{s_t, r_t, \mu_t, \sigma_t, a_t, \hat{q}_t\} \sim RM \right] \quad [6]$$

The *off-PG* formalizes trial-and-error learning; it moves the policy to make actions with better-than-expected returns ($\hat{q}_t > v_{\mathtt{w}}(s_t)$) more likely, and those with worse outcomes ($\hat{q}_t < v_{\mathtt{w}}(s_t)$) less likely. Both Eq. 4 and Eq. 6 involve expectations over the empirical distribution of experiences contained in the RM, which are approximated by Monte Carlo sampling from the N_{RM} most recent experiences $\hat{g}(\mathtt{w}) = \sum_{i=1}^{B} \hat{g}_i(\mathtt{w})$, where B the mini-batch size. Owing to its use of ER and importance sampling, V-RACER and similar algorithms become unstable if the policy $\pi_{\mathtt{w}}$, and the distribution of states that would be visited by $\pi_{\mathtt{w}}$, diverges from the distribution of experiences in the RM. A practical reason for the instability may be the numerically vanishing or exploding importance weights $\pi_{\mathtt{w}}(a_t|s_t)/\mathcal{P}(a_t|\mu_t, \sigma_t)$. ReF-ER is an extended ER procedure which constrains policy changes and increases the accuracy of the gradient estimates by modifying the update rules of the RL algorithm:

$$\hat{g}_t(\mathtt{w}) \leftarrow \begin{cases} \beta \hat{g}_t(\mathtt{w}) - (1-\beta)g_t^D(\mathtt{w}) & \text{if } \frac{1}{C} < \frac{\pi_{\mathtt{w}}(a_t|s_t)}{\mathcal{P}(a_t|\mu_t, \sigma_t)} < C \\ -(1-\beta)g_t^D(\mathtt{w}) & \text{otherwise} \end{cases} \quad [7]$$

here $g_t^D(\mathtt{w}) = \nabla_{\mathtt{w}} D_{KL}(\pi_{\mathtt{w}}(\cdot|s_t) \| \mathcal{P}(\cdot|\mu_t, \sigma_t))$ and $D_{KL}(P \| Q)$ is the Kullback-Leibler divergence measuring the distance between distributions P and Q. Equation 7 modifies the NN gradient by: 1) Rejecting samples whose importance weight is outside of a trust region determined by $C > 1$. 2) Adding a penalization term to attract $\pi_{\mathtt{w}}(a_t|s_t)$ towards prior policies. The coefficient β is iteratively updated to keep a constant fraction $D \in [0, 1]$ of samples in the RM within the trust region:

$$\beta \leftarrow \begin{cases} (1-\eta)\beta & \text{if } n_{far}/N_{RM} > D \\ \beta + (1-\eta)\beta & \text{otherwise} \end{cases} \quad [8]$$

Here n_{far}/N_{RM} is the fraction of the RM with importance weights outside the trust region.

Overview of the training set-up The two most notable hyper-parameters used in our description of the MARL setup are the actuation frequency (determined by Δt_{RL}) and the spatial resolution for the interpolation of the RL actions onto the grid (determined by N_{agents}). Both hyper-parameters serve the purpose of cutting
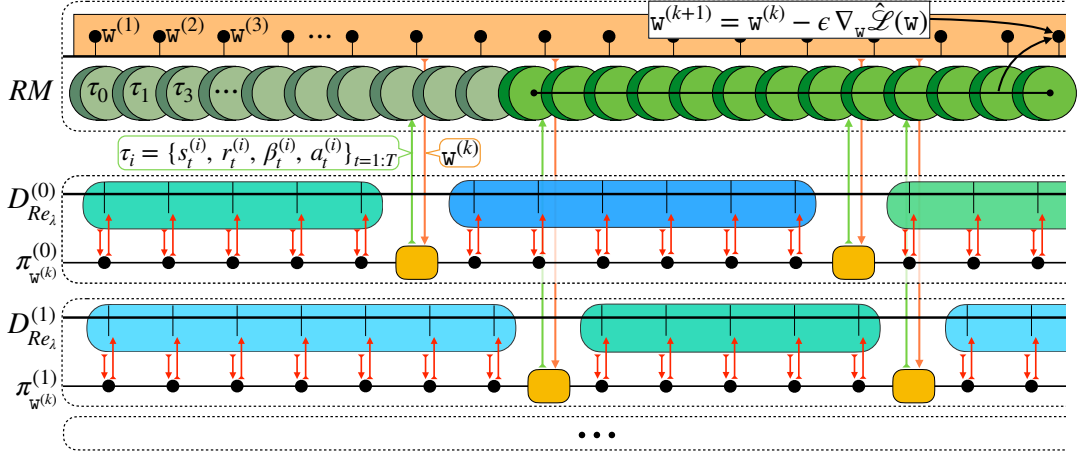
Fig. 5. Schematic description of the training procedure implemented with the `smarties` library. Each dashed line represents a separate process. Multiple worker processes j run LES for randomly sampled $Re_\lambda \in \{65, 76, 88, 103, 120, 140, 163\}$. The Reynolds number determines the dynamics $(D_{Re_\lambda}^{(j)})$ and the number of RL steps per simulation (i.e. the number of times the SGS dissipation coefficient is updated for the entire flow). At the end of each simulation, workers send full episodes τ_i (one per agent in the grid) and receive updated policy parameters. At the top, the diagram outlines the two main tasks performed by the master process: (i) storing the N most recently collected RL steps into a Replay Memory (RM) and (ii) sampling mini-batches from the RM in order to iteratively update the policy parameters $w^{(k)}$ by gradient descent, advancing the update counter k. The policy is updated once every time any worker integrates its simulation over the RL step (i.e. one gradient step per SGS field update).

down the amount of experiences collected during each simulation. The alternative would be to use the policy to compute C_s^2 for each grid-point of the domain and update its value on every simulation time-step. This would produce $\mathcal{O}(10^9)$ experiences per simulation and would make the temporal credit-assignment task (i.e. the RL objective of finding causal correlation between single actions and the observed reward) all the more difficult. The default values $\Delta t_{RL} = \tau_\eta/8$ and $N_{agents} = 4^3$ reduce the number of experiences generated per simulation to $\mathcal{O}(10^5)$. We found that further reducing either the actuation frequency or the number of agents per simulation reduced the model's adaptability and therefore exhibit slightly lower performance.

Each LES is initialized for uniformly sampled $Re_\lambda \in \{65, 76, 88, 103, 120, 140, 163\}$ and a random velocity field synthesized from the target DNS spectrum. The residual-stress tensor τ^R is updated with equation 17 and agents' actions every Δt_{RL}. The LES are interrupted at $T_{end} = 20\tau_I$ (between 750, if $Re_\lambda = 65$, and 1600, if $Re_\lambda = 163$, actions per agent) or if $\|\mathbf{u}\|_\infty > 10^3 u_\eta$, which signals numerical instability. The policy π_w is parameterized by a NN with 2 hidden layers of 64 units each, with tanh activations and skip connections. The NN is initialized as in (47) with small outer weights and bias shifted such that the initial policy is approximately $\pi_{w^{(0)}}(\cdot|s) \approx \mathcal{N}(0.04, 10^{-4})$ and produces Smagorinsky coefficients with small perturbations around $C_s \approx 0.2$. Gradients are computed with Monte Carlo estimates with sample size $B = 512$ from a RM of size $N_{RM} = 10^6$. The parameters are updated with the Adam algorithm (48) with learning rate $\eta = 10^{-5}$. Each training run is advanced for 10^7 policy gradient steps. As discussed in the main text, because we use conventional RL update rules in a multi-agent setting, single parameter updates are imprecise. We found that ReF-ER with hyper-parameters $C = 1.5$ (Eq. 7) and $D = 0.05$ (Eq. 8) to stabilize training. We ran multiple training runs per reward function and whenever we vary the hyper-parameters, but we observe consistent training progress regardless of the initial random seed. The trained policies are evaluated by deterministically setting actions equal to the mean of the Gaussian $a(\mathbf{x}, t) = \mu_w(s(\mathbf{x}, t))$, rather than via sampling, and integrated in time for $100\tau_I$.

Forced Isotropic Turbulence A turbulent flow is *isotropic* when the averaged quantities of the flow are invariant under arbitrary translations and rotations. The flow statistics are independent of space and the mean velocity of the flow is zero. Forced, isotropic turbulence is governed by the incompressible Navier-Stokes equa-

tions,

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\, \mathbf{u} & = -\nabla p + \nabla \cdot (2\nu S) + \mathbf{f} \\ \nabla \cdot \mathbf{u} & = 0 \end{cases} \quad [9]$$

where $S = \frac{1}{2}(\nabla \mathbf{u} + \nabla \mathbf{u}^T)$ is the rate-of-strain tensor. The the turbulent kinetic energy (TKE, the second order statistics of the velocity field) is expressed as:

$$e(\mathbf{x}, t) \equiv \frac{1}{2}\mathbf{u} \cdot \mathbf{u}, \quad K(t) \equiv \frac{1}{2}\langle \mathbf{u} \cdot \mathbf{u} \rangle, \quad [10]$$

where the angle brackets $\langle \cdot \rangle \equiv \frac{1}{\mathcal{V}} \int_{\mathcal{D}} \cdot$ denote an ensemble average over the domain $\mathcal{D}$ with volume $\mathcal{V}$. For a flow with periodic boundary conditions the evolution of the kinetic energy is described as:

$$\frac{dK}{dt} = -\nu \int_{\mathcal{D}} \|\nabla \mathbf{u}\|^2 + \int_{\mathcal{D}} \mathbf{u}\mathbf{f} = -2\nu \langle Z \rangle + \langle \mathbf{u} \cdot \mathbf{f} \rangle \quad [11]$$

where the energy dissipation due to viscosity, is expressed in term of the norm of the vorticity $\boldsymbol{\omega} \equiv \nabla \times \mathbf{u}$ and the *enstrophy* $Z = \frac{1}{2}\omega^2$. This equation clarifies that the vorticity of the flow field is responsible for energy dissipation that can only be conserved if there is a source of energy.

We investigate the behaviour of isotropic turbulence in a statistically stationary state by injecting energy through forcing. In generic flow configurations the role of this forcing is taken up by the large-scale structures and it is assumed that it does not influence smaller scale statistics, which are driven by viscous dissipation. The injected energy is transferred from large-scale motion to smaller scales due to the non-linearity of Navier-Stokes equations. We implement a classic low-wavenumber (low-k) forcing term (49) for isotropic turbulence that is proportional to the local fluid velocity as filtered from its large wave number components:

$$\tilde{\mathbf{f}}(\mathbf{k}, t) \equiv \alpha\, G(\mathbf{k}, k_f)\, \tilde{\mathbf{u}}(\mathbf{k}, t) = \alpha\, \tilde{\mathbf{u}}_<(\mathbf{k}, t), \quad [12]$$

where the tilde symbol denotes a three-dimensional Fourier transform, $G(\mathbf{k}, k_f)$ is a low-pass filter with cutoff wavelength k_f, α a constant, and $\tilde{\mathbf{u}}_<$ is the filtered velocity field. By applying Parseval's theorem, the rate-of-change of energy in the system due to the force is:

$$\langle \mathbf{f} \cdot \mathbf{u} \rangle = \frac{1}{2} \sum_\mathbf{k} \left(\tilde{\mathbf{f}}^* \cdot \tilde{\mathbf{u}} + \tilde{\mathbf{f}} \cdot \tilde{\mathbf{u}}^* \right) = \alpha \sum_\mathbf{k} \tilde{\mathbf{u}}_<^2 = 2\alpha K_<. \quad [13]$$

Here, $K_<$ is the kinetic energy of the filtered field. We set $\alpha = \epsilon/2K_<$ and $k_f = 4\pi/L$, meaning that we simulate a time-constant rate of energy injection ϵ which forces only the seven lowest modes
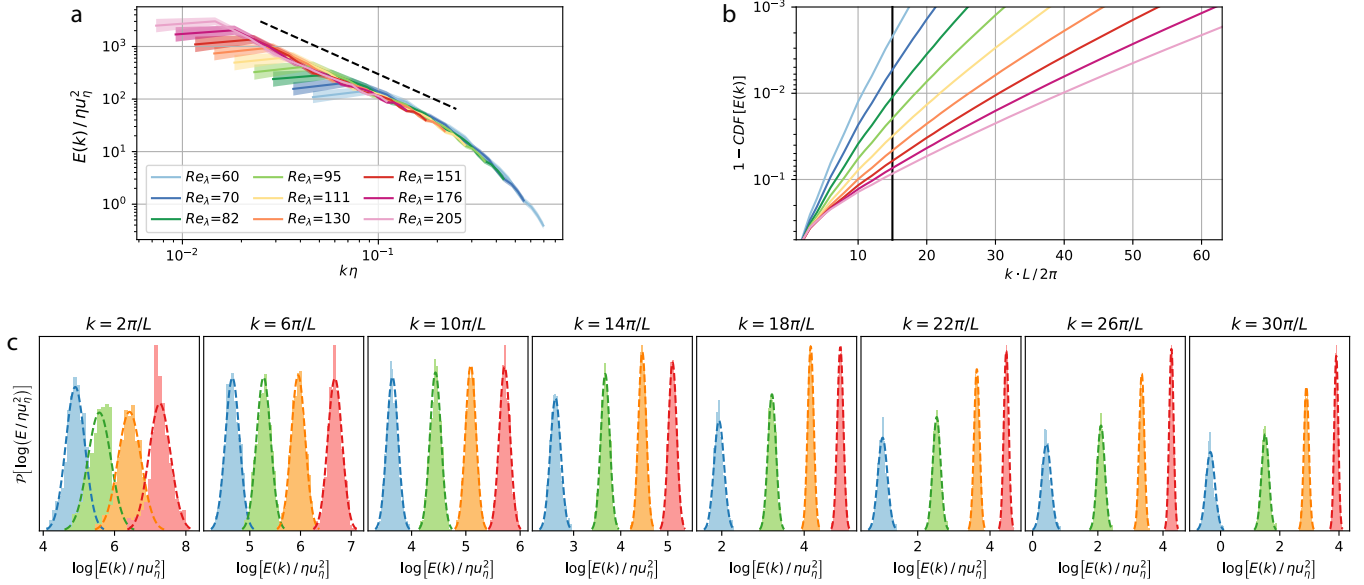
Fig. 6. Statistical properties of DNS simulations of forced isotropic turbulence. a, Time averaged energy spectra, and intervals of one standard deviation for log-increments of $Re_\lambda \in [60, 205]$ compared to Kolmogorov's spectrum $\propto k^{-5/3}$ (dashed line). **b**, Cumulative fraction of the TKE contained up to mode k. The black vertical line corresponds to the Nyquist frequency for the grid size ($N = 32^3$) used for all LES considered throughout this study. **c**, Histograms of single modes of the energy spectrum for $Re_\lambda = 65$ (blue), 88 (green), 110 (orange), and 163 (red) compared to a Gaussian fit.

of the energy spectrum. The constant injection rate is counter-balanced by the viscous dissipation $\epsilon_{visc} = 2\nu \langle Z \rangle$, the dissipation due to the numerical errors ϵ_{num}, and, by a subgrid-scale (SGS) model of turbulence (ϵ_{sgs}, when it is employed - see Sec. 1). When the statistics of the flow reach steady state, the time-averaged total rate of energy dissipation $\epsilon_{tot} = \epsilon_{visc} + \epsilon_{num} + \epsilon_{sgs}$ is equal to the rate of energy injection ϵ.

The Characteristic Scales of Turbulence Turbulent flows are characterized by a large separation in temporal and spatial scales and long-term dynamics. These scales can be estimated by means of dimensional analysis, and can be used to characterize turbulent flows. At the *Kolmogorov scales* energy is dissipated into heat: $\eta = \left(\nu^3/\epsilon\right)^{1/4}$, $\tau_\eta = (\nu/\epsilon)^{1/2}$, $u_\eta = (\epsilon\nu)^{1/4}$. These quantities are independent of large-scale effects including boundary conditions or external forcing. The *integral scales* are the scales of the largest eddies of the flow: $l_I = \frac{3\pi}{4K} \int_0^\infty \frac{\tilde{E}(k)}{k} dk$, $\tau_I = \frac{l_I}{\sqrt{2K/3}}$. The Taylor-Reynolds number is used to characterize flows with zero mean bulk velocity: $Re_\lambda = K\sqrt{20/(3\nu\epsilon)}$.

Under the assumptions of isotropic flow we study the statistical properties of turbulence in Fourier space. We analyze quantities computed from simulations at statistically steady state and we omit the temporal dependencies. The energy spectrum is $\tilde{E}(k) \equiv \frac{1}{2}\tilde{u}^2(k)$. Kolmogorov's theory of turbulence predicts the well-known $-\frac{5}{3}$ spectrum (i.e. $\tilde{E}(k) \propto \epsilon^{2/3}k^{-5/3}$) for the turbulent energy in the inertial range $k_I \ll k \ll k_\eta$.

Direct Numerical Simulations (DNS) Data from DNS serve as reference for the SGS models and as targets for creating training rewards for the RL agents. The DNS are carried out on a uniform grid of size 512^3 for a periodic cubic domain $(2\pi)^3$. The solver is based on finite differences, third-order upwind for advection and second-order centered differences for diffusion, and pressure projection (50). Time stepping is performed with second-order explicit Runge-Kutta with variable integration step-size determined with a Courant–Friedrichs–Lewy (CFL) coefficient $CFL = 0.1$. We performed DNS for Taylor-Reynolds numbers in log increments between $Re_\lambda \in [60, 205]$ (Fig. 6**a,b**).

The initial velocity field is synthesized by generating a distribution of random Fourier coefficients matching a radial target spectrum $\tilde{E}(k)$ (51): $\tilde{E}(k) = c_k \, \epsilon^{2/3} k^{-5/3} \, f_L(kL) \, f_\eta(k\eta)$, where $f_l(kl_I)$ and

$f_\eta(k\eta)$ determine the spectrum in the integral- and the dissipation-ranges respectively (3). The choice of initial spectrum determines how quickly the simulation reaches statistical steady state, at which point Re_λ fluctuates around a constant value. The time-averaged quantities (Fig. 6) are computed from 20 independent DNS with measurements taken every τ_η. Each DNS lasts 20 τ_I and the initial 10 τ_I are not included in the measurements, which found to be ample time to avoid the initial transient. Figure 6**c** shows that the distribution of energy content for each mode $\tilde{E}(k)$ is well approximated by a log-normal distribution such that $\log E_{DNS}^{Re_\lambda} \sim \mathcal{N}\left(\mu_{DNS}^{Re_\lambda}, \Sigma_{DNS}^{Re_\lambda}\right)$, where $\mu_{DNS}^{Re_\lambda}$ is the empirical average of the log-energy spectrum for a given Re_λ and $\Sigma_{DNS}^{Re_\lambda}$ is its covariance matrix.

Large-Eddy Simulations (LES) LES (7) resolve the large scale dynamics of turbulence and model their interaction with the sub grid-scales (SGS). The flow field $\bar{u}$ on the grid is viewed as the result of filtering out the residual small-scales of a latent velocity field u. The filtered Navier-Stokes equation for the field $\bar{u}$ reads:

$$\frac{\partial \bar{u}}{\partial t} + (\bar{u} \cdot \nabla) \bar{u} = -\nabla \bar{p} + \nabla \cdot \left(2\nu\bar{S} - \tau^R\right) + \bar{f} \qquad [14]$$

here, the *residual-stress tensor* τ^R encloses the interaction with the unresolved scales:

$$\tau^R = \overline{u \otimes u} - \bar{u} \otimes \bar{u}. \qquad [15]$$

Closure equations are used to model the sub grid-scale motions represented by $\overline{u \otimes u}$.

The Classic Smagorinsky Model (SSM) The SSM (8) is a linear eddy-viscosity model that relates the residual stress-tensor to the filtered rate of strain

$$\tau^R - \frac{1}{3}\text{tr}\left(\tau^R\right) = -2\,\nu_t\,\bar{S}, \qquad [16]$$

$$\nu_t = (C_s\Delta)^2 \, \|\bar{S}\|, \qquad [17]$$

where Δ is the grid size and C_s is a constant. This model has been shown to perform reasonably well for isotropic turbulence and wall-bounded turbulence. Eq. 16 models energy transfer from the filtered motions to the residual motions proportional to the turbulent eddy-viscosity ν_t. The main drawback of this model is that the constant C_s has to be manually tuned.

The Dynamic Smagorinsky Model (DSM) The DSM (9) computes the parameter $C_s(x, t)$ as a function of space and time.

DSM's dynamic model is obtained by filtering equation 14 a second time with a so-called test filter of size $\widehat{\Delta} > \Delta$. The resolved-stress tensor $\mathcal{L}$ is defined by the Germano identity:

$$\mathcal{L}_{\overline{\boldsymbol{u}}} = \widehat{\overline{\boldsymbol{u}} \otimes \overline{\boldsymbol{u}}} - \widehat{\overline{\boldsymbol{u}}} \otimes \widehat{\overline{\boldsymbol{u}}} = T^R - \widehat{\tau^R}, \qquad [18]$$

where $T^R = \widehat{\overline{\boldsymbol{u} \otimes \boldsymbol{u}}} - \widehat{\overline{\boldsymbol{u}}} \otimes \widehat{\overline{\boldsymbol{u}}}$ is the residual-stress tensor for the test filter width $\widehat{\Delta}$, and $\widehat{\tau^R}$ is the test-filtered residual stress tensor for the grid size Δ (Eq. 15). If both residual stresses are approximated by a Smagorinsky model, the Germano identity becomes:

$$\mathcal{L}_{\overline{\boldsymbol{u}}} \approx 2\, C_s^2(\boldsymbol{x}, t)\, \Delta^2 \left[\widehat{\|\widehat{\overline{S}}\|\widehat{\overline{S}}} - \frac{\widehat{\Delta}^2}{\Delta^2} \|\widehat{\overline{S}}\|\widehat{\overline{S}} \right]. \qquad [19]$$

The dynamic Smagorinsky parameter (Eq. 19) forms an over-determined system for $C_s^2(\boldsymbol{x}, t)$, whose least-squares solution is (10):

$$C_s^2(\boldsymbol{x}, t) = \frac{\langle \mathcal{L}_{\overline{\boldsymbol{u}}}, \mathcal{M} \rangle_F}{2\Delta^2 \, \|\mathcal{M}\|^2}, \qquad [20]$$

where $\mathcal{M} = \widehat{\|\widehat{\overline{S}}\|\widehat{\overline{S}}} - (\widehat{\Delta}/\Delta)^2 \|\widehat{\overline{S}}\|\widehat{\overline{S}}$, and $\langle \cdot \rangle_F$ is the Frobenius product. Because the dynamic coefficient may take negative values, which represents energy transfer from the unresolved to the resolved scales, C_s^2 is clipped to positive values for numerical stability.

The fraction of TKE contained in the unresolved scales increases with Re_λ and decreases with the grid size (Fig. 6**b**). For all LES considered in this study we employ a grid of size $N = 32^3$ and time-stepping coefficient $CFL = 0.1$. For the higer Re_λ, the SGS model accounts for up to 10% of the total TKE. We employ second-order centered discretization for the advection and the initial conditions for the velocity field are synthesized from the time-averaged DNS spectrum at the same Re_λ (51). When reporting results from SSM simulation, we imply the Smagorinsky constant C_s resulting from line-search optimization. LES statistics are computed from simulations up to $t = 100\tau_I$, disregarding the initial $10\tau_I$ time units. For the DSM procedure we employ an uniform box test-filter of width $\widehat{\Delta} = 2\Delta$.

MARL models We defined two MARL SGS models by the reward function they optimize. Both reward functions have the form $r(\boldsymbol{x}^{(i)}, t) = r^{grid}(t) + r'(\boldsymbol{x}^{(i)}, t)$. The base reward is a measure of the distance from the target DNS spectrum, derived from the regularized log-likelihood (Eq. 2):

$$r^{grid}(t) = \exp\left[-\sqrt{-\langle \widetilde{LL} \rangle(t)} \right]. \qquad [21]$$

This regularized distance is preferred because a reward directly proportional to the probability $\mathcal{P}(\tilde{E}(t)|E_{DNS}^{Re_\lambda})$ (Eq. 3) quickly vanishes to zero for imperfect SGS models and therefore yields too flat an optimization landscape. The average LES spectrum is computed with an exponential moving average with effective window Δt_{RL}:

$$\langle \widetilde{LL} \rangle(t) = \langle \widetilde{LL} \rangle(t - \delta t) + \frac{\delta t}{\Delta t_{RL}} \left(\widetilde{LL}(\tilde{E}(t)|E_{DNS}) - \langle \widetilde{LL} \rangle(t - \delta t) \right) \qquad [22]$$

The reward r^G adds a local term to reward actions that satisfy the Germano identity (Eq. 18):

$$r^G(\boldsymbol{x}, t) = r^{grid}(t) - \frac{1}{u_\eta^4} \|\mathcal{L}_{\overline{\boldsymbol{u}}}(\boldsymbol{x}, t) - T^R(\boldsymbol{x}, t) + \widehat{\tau^R(\boldsymbol{x}, t)}\|^2 \qquad [23]$$

here the coefficient u_η^4 is introduced for non-dimensionalization. The reward r^{LL} further rewards matching the DNS spectra:

$$r^{LL}(t) = r^{grid}(t) + \frac{\tau_\eta}{\Delta t_{RL}} \left[\langle \widetilde{LL} \rangle(t) - \langle \widetilde{LL} \rangle(t - \Delta t_{RL}) \right] \qquad [24]$$

This can be interpreted as a non-dimensional derivative of the log-likelihood over the RL step, or a measure of the contribution of each round of SGS model update to the instantaneous accuracy of the LES.

Author contributions. G.N. and P.K. designed the research; G.N. and H.L. wrote the simulation software, G.N., H.L. and P.K. carried out the research; G.N. and P.K. wrote the paper.

1. AN Kolmogorov, The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. *Dokl. Akad. Nauk S.S.S.R.* **30**, 299–301 (1941).
2. GI Taylor, Statistical theory of turbulence: Parts i-ii. *Proc. Royal Soc. London. Ser. A-Mathematical Phys. Sci.* **151**, 444–454 (1935).
3. SB Pope, Turbulent flows (2001).
4. P Moin, K Mahesh, Direct numerical simulation: A tool in turbulence research. *Annu. Rev. Fluid Mech.* **30**, 539–578 (1998).
5. RD Moser, J Kim, NN Mansour, Direct numerical simulation of turbulent channel flow up to re $\tau = 590$. *Phys. fluids* **11**, 943–945 (1999).
6. PA Durbin, Some recent developments in turbulence closure modeling. *Annu. Rev. Fluid Mech.* **50**, 77–103 (2018).
7. A Leonard, , et al., Energy cascade in large-eddy simulations of turbulent fluid flows. *Adv. Geophys. A* **18**, 237–248 (1974).
8. J Smagorinsky, General circulation experiments with the primitive equations: I. the basic experiment. *Mon. weather review* **91**, 99–164 (1963).
9. M Germano, U Piomelli, P Moin, WH Cabot, A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids A: Fluid Dyn.* **3**, 1760–1765 (1991).
10. DK Lilly, A proposed modification of the germano subgrid-scale closure method. *Phys. Fluids A: Fluid Dyn.* **4**, 633–635 (1992).
11. C Lee, J Kim, D Babcock, R Goodman, Application of neural networks to turbulence control for drag reduction. *Phys. Fluids* **9**, 1740–1747 (1997).
12. M Milano, P Koumoutsakos, Neural network modeling for near wall turbulent flow. *J. Comput. Phys.* **182**, 1–26 (2002).
13. K Duraisamy, G Iaccarino, H Xiao, Turbulence modeling in the age of data. *Annu. Rev. Fluid Mech.* **51**, 357–377 (2019).
14. F Sarghini, G De Felice, S Santini, Neural networks based subgrid scale modeling in large eddy simulations. *Comput. & fluids* **32**, 97–108 (2003).
15. M Gamahara, Y Hattori, Searching for turbulence models by artificial neural network. *Phys. Rev. Fluids* **2**, 054604 (2017).
16. C Xie, J Wang, H Li, M Wan, S Chen, Artificial neural network mixed model for large eddy simulation of compressible isotropic turbulence. *Phys. Fluids* **31**, 085112 (2019).
17. A Vollant, G Balarac, C Corre, Subgrid-scale scalar flux modelling based on optimal estimation theory and machine-learning procedures. *J. Turbul.* **18**, 854–878 (2017).
18. S Hickel, S Franz, N Adams, P Koumoutsakos, Optimization of an implicit subgrid-scale model for les in *Proceedings of the 21st International Congress of Theoretical and Applied Mechanics, Warsaw, Poland.* (2004).
19. R Maulik, O San, A neural network approach for the blind deconvolution of turbulent flows. *J. Fluid Mech.* **831**, 151–181 (2017).

20. J Sirignano, JF MacArt, JB Freund, Dpm: A deep learning pde augmentation method with application to large-eddy simulation. *J. Comput. Phys.* **423**, 109811 (2020).

21. JL Wu, H Xiao, E Paterson, Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Phys. Rev. Fluids* **3**, 074602 (2018).

22. B Nadiga, D Livescu, Instability of the perfect subgrid model in implicit-filtering large eddy simulation of geostrophic turbulence. *Phys. Rev. E* **75**, 046303 (2007).

23. A Beck, D Flad, CD Munz, Deep neural networks for data-driven les closure models. *J. Comput. Phys.* **398**, 108910 (2019).

24. RS Sutton, AG Barto, *Reinforcement learning: An introduction*. (MIT press), (2018).

25. V Mnih, et al., Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015).

26. D Silver, et al., Mastering the game of go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).

27. S Levine, C Finn, T Darrell, P Abbeel, End-to-end training of deep visuomotor policies. *The J. Mach. Learn. Res.* **17**, 1334–1373 (2016).

28. Iea Akkaya, Solving Rubik's Cube with a Robot Hand. *arXiv preprint* **1910.07113**, arXiv:1910.07113 (2019).

29. P Garnier, et al., A review on deep reinforcement learning for fluid mechanics. *arXiv preprint* **1908.04127** (2019).

30. M Gazzola, B Hejazialhosseini, P Koumoutsakos, Reinforcement learning and wavelet adapted vortex methods for simulations of self-propelled swimmers. *SIAM J. on Sci. Comput.* **36**, B622–B639 (2014).

31. G Reddy, A Celani, TJ Sejnowski, M Vergassola, Learning to soar in turbulent environments. *Proc. Natl. Acad. Sci.* **113**, E4877–E4884 (2016).

32. G Novati, et al., Synchronisation through learning for two self-propelled swimmers. *Bioinspiration & Biomimetics* **12**, 036001 (2017).

33. S Verma, G Novati, P Koumoutsakos, Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proc. Natl. Acad. Sci.* **115**, 5849–5854 (2018).

34. V Belus, et al., Exploiting locality and translational invariance to design effective deep reinforcement learning control of the 1-dimensional unstable falling liquid film. *AIP Adv.* **9**, 125014 (2019).

35. L Biferale, F Bonaccorso, M Buzzicotti, P Clark Di Leoni, K Gustavsson, Zermelo's problem: Optimal point-to-point navigation in 2d turbulent flows using reinforcement learning. *Chaos: An Interdiscip. J. Nonlinear Sci.* **29**, 103138 (2019).

36. G Novati, L Mahadevan, P Koumoutsakos, Controlled gliding and perching through deep-reinforcement-learning. *Phys. Rev. Fluids* **4**, 093902 (2019).

37. V François-Lavet, et al., An introduction to deep reinforcement learning. *Foundations Trends Mach. Learn.* **11**, 219–354 (2018).

38. J Ling, A Kurzawski, J Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166 (2016).

39. S Pope, A more general effective-viscosity hypothesis. *J. Fluid Mech.* **72**, 331–340 (1975).

40. L Buşoniu, R Babuška, B De Schutter, Multi-agent reinforcement learning: An overview in *Innovations in multi-agent systems and applications-1*. (Springer), pp. 183–221 (2010).

41. G Novati, P Koumoutsakos, Remember and forget for experience replay in *Proceedings of the 36th International Conference on Machine Learning*. (2019).

42. L Lin, Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach. learning* **8**, 69–97 (1992).

43. J Meyers, BJ Geurts, M Baelmans, Database analysis of errors in large-eddy simulation. *Phys. Fluids* **15**, 2740–2755 (2003).

44. Y Zhiyin, Large-eddy simulation: Past, present and the future. *Chin. journal Aeronaut.* **28**, 11–24 (2015).

45. T Degris, M White, RS Sutton, Off-policy actor-critic in *Proceedings of the 36th International Conference on Machine Learning*. (2012).

46. R Munos, T Stepleton, A Harutyunyan, M Bellemare, Safe and efficient off-policy reinforcement learning in *Advances in Neural Information Processing Systems*. pp. 1054–1062 (2016).

47. X Glorot, Y Bengio, Understanding the difficulty of training deep feedforward neural networks in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. pp. 249–256 (2010).

48. DP Kingma, J Ba, Adam: A method for stochastic optimization in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. (2014).

49. S Ghosal, TS Lund, P Moin, K Akselvoll, A dynamic localization model for large-eddy simulation of turbulent flows. *J. fluid mechanics* **286**, 229–255 (1995).

50. AJ Chorin, A numerical method for solving incompressible viscous flow problems. *J. computational physics* **2**, 12–26 (1967).

51. RS Rogallo, P Moin, Numerical simulation of turbulent flows. *Annu. review fluid mechanics* **16**, 99–137 (1984).