nature machine intelligence

ARTICLES

https://doi.org/10.1038/s42256-020-00272-0

Automating turbulence modelling by multi-agent reinforcement learning

Guido Novati[1], Hugues Lascombes de Laroussilhe[1,2] and Petros Koumoutsakos[1,3] ✉

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 endeavour with promising results. However, all efforts have focused on supervised learning, which is difficult to generalize beyond training data. Here we introduce multi-agent reinforcement learning as an automated discovery tool of turbulence models. We demonstrate the potential of this approach on large-eddy simulations of isotropic turbulence, using the recovery of statistical properties of direct numerical simulations as a reward. 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 subgrid-scale physics. Results obtained with multi-agent reinforcement learning algorithms based on experience replay compare favourably with established modelling approaches. Moreover, we show that the learned turbulence models generalize across grid sizes and flow conditions.

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 past sixty years we have increasingly relied on such predictions for 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 past 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 insights into the physics of turbulence that can lead in turn to effective turbulence modelling. 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 on the subgrid-scale (SGS), unresolved, physics are modelled. In LES, classic approaches to the explicit modelling of SGS stresses include the standard[8] and the dynamic Smagorinsky model[9,10]. In the past 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 fuelled 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 neural network (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 (for example 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

[1]Computational Science and Engineering Laboratory, ETH Zürich, Zurich, Switzerland. [2]Present address: Institute for Computational Science, University of Zurich, Zurich, Switzerland. [3]Present address: School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. ✉e-mail: petros@ethz.ch

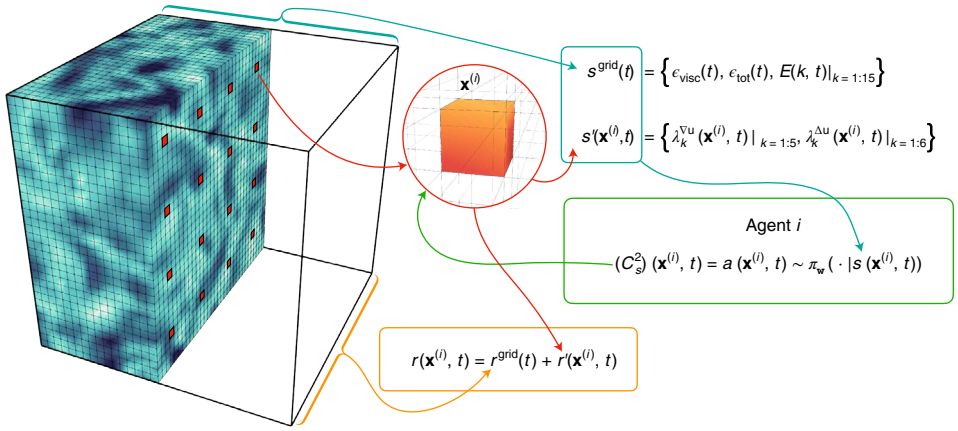

Fig. 1 | Schematic of the integration of MARL with the flow solver. The agents (located at the red blocks) compute the SGS dissipation coefficient C_s^2 for each grid point of the simulation by sampling a shared control policy $\pi_w(\cdot|s(\mathbf{x},t))$. The state $s(\mathbf{x},t)$ of agent i at time t is defined by both local variables (that is, $s'(\mathbf{x},t)$, 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 (that is, $s^{\text{grid}}(t)$, 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.

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 analyse 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 modelling errors. Moreover, the performance of an 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 subgrid-scale modelling

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 NNs to represent the policy ($\pi_w(\cdot|s)$, 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 a considerable effect on the computational efficiency of the resulting model. As an example, following the common practice in video games[25], the state 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 simulation domain (Fig. 1). Each agent (i, with spatial coordinate $\mathbf{x}^{(i)}$) performs a localized action based on information about the state of the flow field $s(\mathbf{x}^{(i)},t) \in \mathbb{R}^{\dim s}$, 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 five invariants[39] of the gradient ($\lambda_k^{\nabla \mathbf{u}}$) and the six invariants of the Hessian of the velocity field ($\lambda_k^{\Delta \mathbf{u}}$). These are computed at the agent's 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 ($\epsilon_{\text{visc}}/\epsilon$) and the total dissipation ($\epsilon_{\text{tot}}/\epsilon$) 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 $N_{\mathrm{Nyquist}}=15$, 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 $C_s^2(\mathbf{x},t)$ for the whole flow and integrating the LES in time for Δt_{RL}, until $t=T_{\mathrm{end}}$ or any numerical instability arises. At the start of every turn, each agent i measures its state and selects an action $a(\mathbf{x}^{(i)},t)\in\mathbb{R}$ by sampling a Gaussian policy: $a(\mathbf{x}^{(i)},t)\sim\pi_w(\cdot\mid s(\mathbf{x}^{(i)},t))\equiv\mathcal{N}[\mu_w(s(\mathbf{x}^{(i)},t)),\ \sigma_w^2(s(\mathbf{x}^{(i)},t))]$. The actions are used to compute $C_s^2(\mathbf{x},t)$ for each grid point by linear interpolation:

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

where $x_j^{(i)}$ is the j-th Cartesian component of the position vector of agent i, and $\Delta_{\mathrm{agents}}=2\pi/\sqrt{[}3]N_{\mathrm{agents}}$ is the distance between agents. If $N_{\mathrm{agents}}=N$, 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 increase 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 w that maximize the expected sum of rewards over each simulation $J(w)=\mathbb{E}_{\pi_w}\left[\sum_{t=1}^{T_{\mathrm{end}}}r_t\right]$. In the present study, the parameters w are shared by all agents. Their aggregate experiences are collected in a shared dataset and used to compute updates according to ReF-ER (Fig. 2). We define reward functions with the objective of obtaining policies π_w that yield stable LES and exhibit statistical properties that closely match those of DNS. We find that, for DNS of isotropic turbulence, the distribution of energy contained in each mode of the spectrum $E(k)$ is well approximated by a log-normal distribution (see Methods and Fig. 3) such that $\log \mathrm{E}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}\sim\mathcal{N}(\mu_{\mathrm{DNS}}^{\mathrm{Re}_\lambda},\Sigma_{\mathrm{DNS}}^{\mathrm{Re}_\lambda})$, where $\mu_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}$ is the average log-energy spectrum for a given Re_λ and $\Sigma_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}$ is its covariance matrix. When comparing SGS models and formulating objective functions, we rely on a regularized log-likelihood:

$$\widetilde{\mathrm{LL}}(E_{\mathrm{LES}}^{\mathrm{Re}_\lambda}|E_{\mathrm{DNS}}^{\mathrm{Re}_\lambda})=\log\mathcal{P}(\mathrm{E}_{\mathrm{LES}}^{\mathrm{Re}_\lambda}|\mathrm{E}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda})/\mathrm{N}_{\mathrm{Nyquist}}\qquad(2)$$

Here the probability metric is

$$\mathcal{P}(E_{\mathrm{LES}}^{\mathrm{Re}_\lambda}|E_{\mathrm{DNS}}^{\mathrm{Re}_\lambda})\propto\exp\left[-\tfrac{1}{2}\left(\log\mathrm{E}_{\mathrm{LES}}^{\mathrm{Re}_\lambda}-\overline{\mu}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}\right)^T\left(\overline{\Sigma}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}\right)^{-1}\left(\log\mathrm{E}_{\mathrm{LES}}^{\mathrm{Re}_\lambda}-\overline{\mu}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}\right)\right]\qquad(3)$$

with E_{LES} the LES energy spectrum, and $\overline{\mu}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}$ and $\overline{\Sigma}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}$ the target statistics up to N_{Nyquist}.

We consider two SGS models derived from MARL, each corresponding to a reward function of the form $r(\mathbf{x}^{(i)},t)$. The first (π_w^G) is defined by rewards $r^G(\mathbf{x}^{(i)},t)$ based on the error in the Germano identity[9], which states that the sum of resolved and modelled contributions to the SGS stress tensor should be independent of LES resolution. The second (π_w^{LL}) is defined by rewards $r^{\mathrm{LL}}(\mathbf{x}^{(i)},t)$ that 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 modelling 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 4 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 (for example $\mathrm{Re}_\lambda=65$) the SGS model is barely able to represent the flow features of DNS. For Reynolds numbers beyond $\mathrm{Re}_\lambda=111$ 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 modelled.

A natural way to assess whether the turbulence model accurately reproduces the energy transfer to SGS motions is through the energy spectra. In Fig. 5a, 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 normalization with the mean and standard deviation of the energy computed through DNS (Fig. 5b). This measure quantifies the contributions of individual modes to the objective log-likelihood (equation (2)). A perfect SGS model would produce a spectrum with time-averaged $E_{\mathrm{LES}}^{\mathrm{Re}_\lambda}(k)$ 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 π_w^{LL} and π_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 numerical scales of the flow quantities, and of the RL state components, vary with Re_λ. Therefore, the results for $\mathrm{Re}_\lambda=82$, 111 and 151 measure the MARL model accuracy for dynamical scales that are interposed with the training ones, while the results for $\mathrm{Re}_\lambda=60$, 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, π_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. 5c, which shows the empirical distribution of Smagorinsky coefficients chosen by the SGS models. While outwardly DSM and π_w^G minimize the same relation, π_w^G introduces much more artificial viscosity.

The policy π_w^{LL}, which directly maximizes the similarity between quantity of interest (that is energy spectra) obtained by MARL and those of DNS, produces the SGS model of the highest quality.

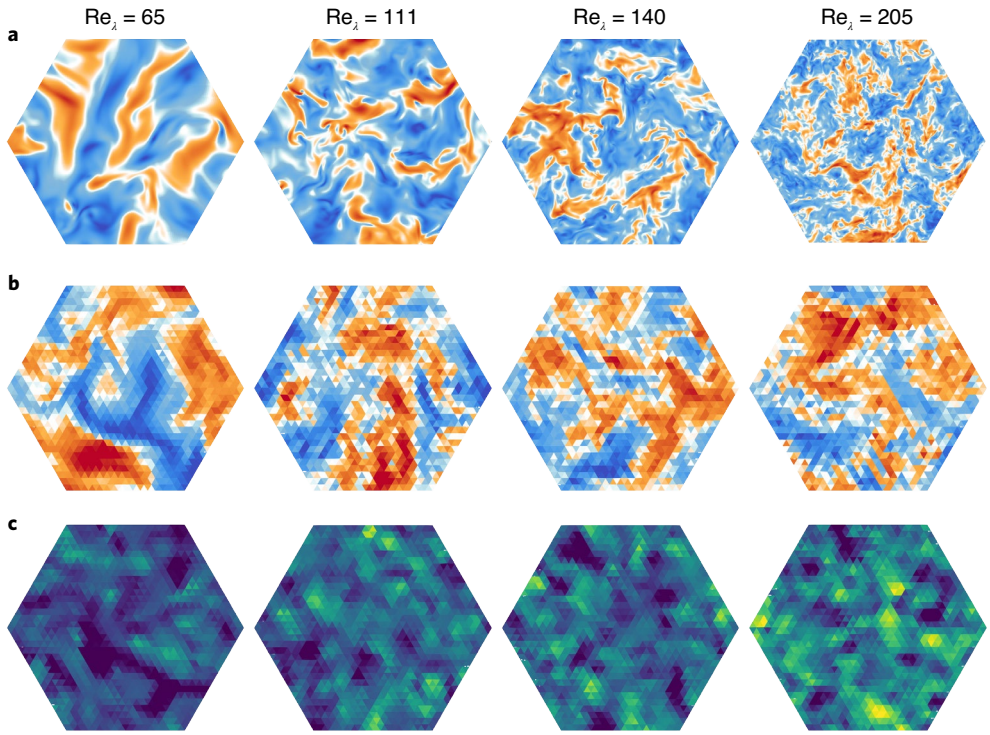

Fig. 2 | Visualizations of simulations of isotropic turbulence. a,**b**, Representative contours of momentum flux across a diagonal slice ($x+y+z=0$) of the cubical domain (**u**·**n**, blue and orange indicate negative and positive fluxes) for DNS of isotropic turbulence with resolution $1{,}024^3$ (**a**) and for LES with resolution 32^3 and SGS modelling with a RL policy trained for r^{LL} (**b**). **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).

While its accuracy is similar to that of DSM for lower values of Re_λ, π_w^{LL} avoids energy build-up and remains stable up to $\mathrm{Re}_\lambda = 205$, well beyond the maximum training $\mathrm{Re}_\lambda = 163$. 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 π_w^{LL} to a policy fitted exclusively for $\mathrm{Re}_\lambda = 111$ ($\pi_w^{\mathrm{LL},111}$). Figure 5b shows the specialized policy to have comparable accuracy at $\mathrm{Re}_\lambda = 111$, 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. 5c, we observe that π_w^{LL} achieves its accuracy by producing a narrower distribution of C_s^2. In this respect, π_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 π_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 modelling errors (that is 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 may inadvertently sacrifice the physical soundness of other quantities in order to maximize its objective function. Fig. 5d 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 π_w^{LL} and SSM in order to maintain stability. Fig. 5e 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 behaviour $S^2(r) \propto r^{\zeta(p)}$ with coefficient depending only on the energy flux ϵ (refs. [1,3]). As expected from DNS, the scaling of $S^2(r)$ approaches a constant value regardless of Re_λ. In Fig. 6a–d we compare the total kinetic energy, 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 $\mathrm{Re}_\lambda = 205$ the stable SGS models dissipate approximately 10 times more energy than viscous dissipation, which underlines the crucial role of turbulence modelling. Up to the point of instability at $\mathrm{Re}_\lambda \approx 100$, DSM yields a good estimate for the kinetic energy 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 π_w^{LL}.

Finally, we evaluate MARL across grid resolutions. Because of the design of the MARL framework, the policy π_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. 6e 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

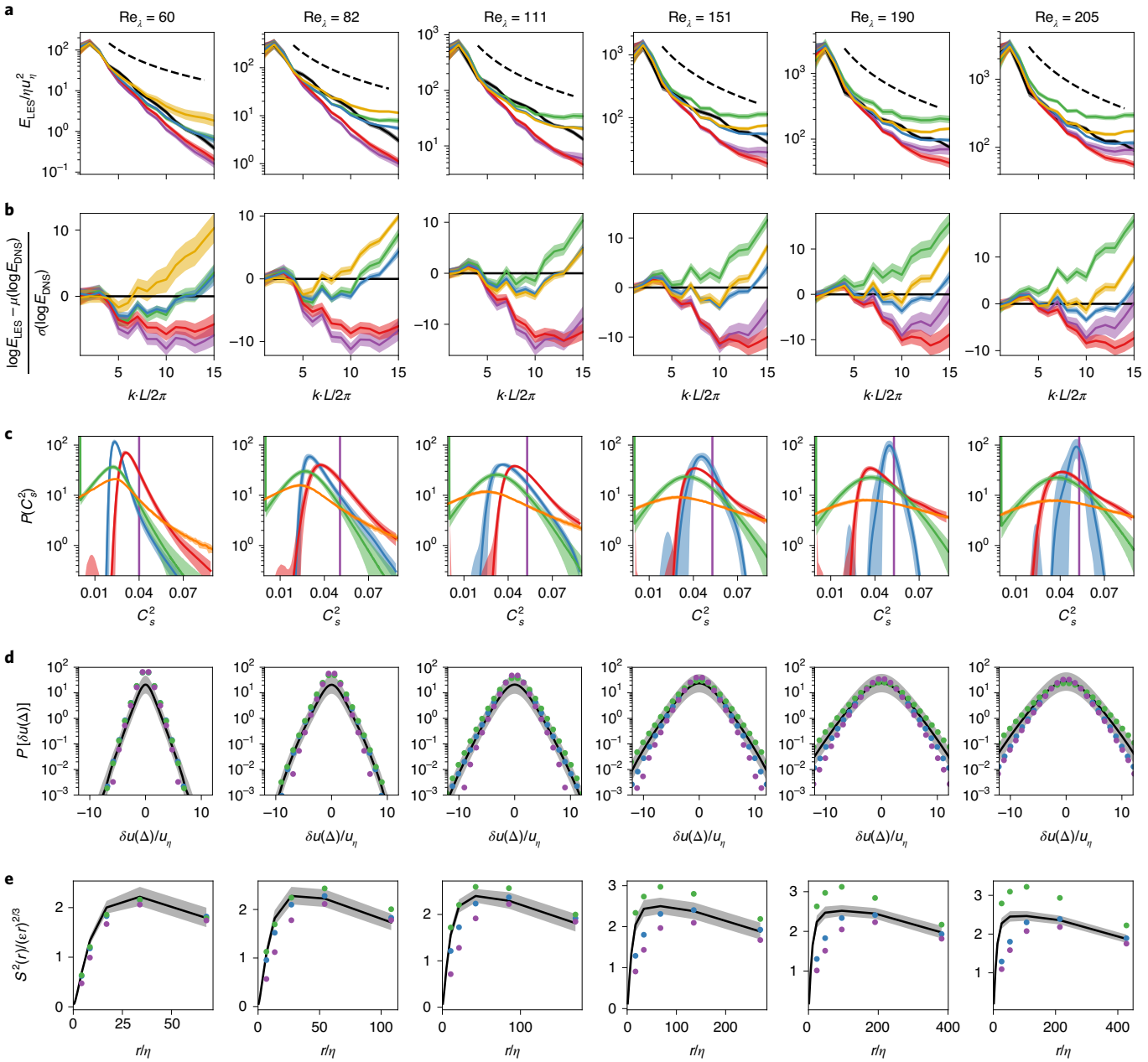

Fig. 3 | Comparison of SGS models for values of Re$_\lambda$ that were not included during the MARL training. a, Energy spectra for DNS (solid black line), SSM (purple), DSM (green), MARL policy π_w^{LL} (blue), MARL policy π_w^{G} (red) and MARL policy $\pi_w^{LL,111}$ (yellow) trained exclusively from data for Re$_\lambda$ = 111, compared with the −5/3 Kolmogorov scaling (dashed line). **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 colours) compared to the SGS dissipation (orange) coefficient computed from DNS filtered to LES resolution. **d**,**e**, 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 (**d**) and second-order velocity structure function $S^2(r) = \langle \delta u(r)^2 \rangle$ (**e**) for π_w^{LL} (blue dots), DSM (green) and SSM (purple) compared to DNS. In all cases, lines represent averages and contours represent intervals of one standard deviation.

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 π_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, π_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$_\lambda$.

Discussion

This paper introduces MARL to automate the discovery of closure models in simulations of turbulent flows. We demonstrate the feasibility and potential of this approach on LES of forced isotropic turbulence. MARL develops the SGS closure as a control policy enacted by cooperating agents. The agents are incorporated into the flow solver, observe local (for example, invariants of the velocity gradient) as well as global (for example, the energy spectrum) flow quantities, and accordingly compute SGS residual-stresses through the Smagorinsky[8] formulation. The ReF-ER method is instrumental

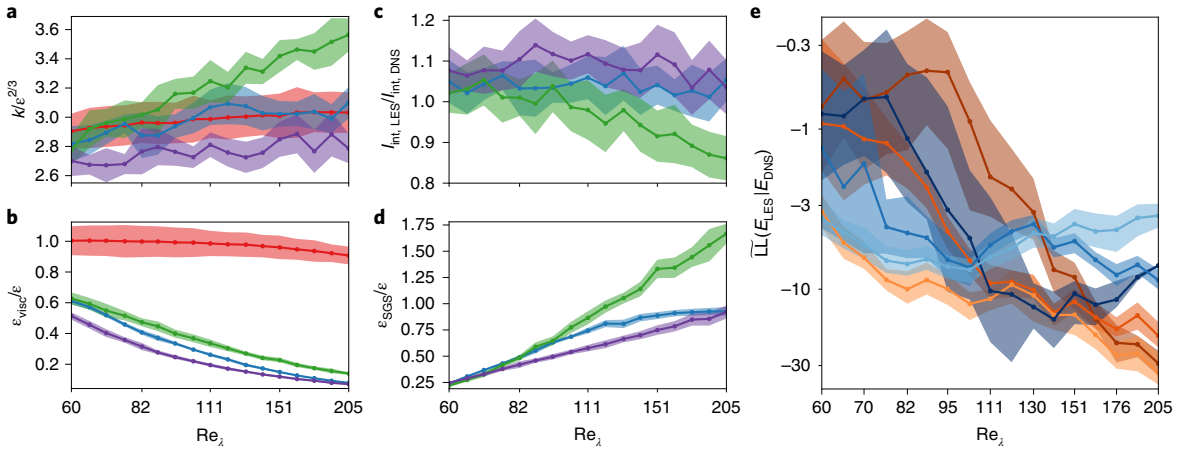

Fig. 4 | Measurements of the reliability of the MARL model beyond the training objective. a–d, Integral properties of LES: turbulent kinetic energy (**a**), integral length scale (**b**), ratio of viscous dissipation to energy injection (**c**) and ratio of SGS dissipation to energy injection (**d**). The remaining component of energy dissipation is due to numerical discretization. SSM (purple), DSM (green), MARL policy π_w^{LL} (blue) and DNS simulation (red) when applicable. **e**, Accuracy across grid resolutions measured as log-likelihood of LES spectra with respect to DNS statistics (equation (2)) for DSM at resolutions $N=32^3$ (yellow), $N=64^3$ (orange) $N=128^3$ (brown) and MARL policy π_w^{LL} at resolutions $N=32^3$ (light blue), $N=64^3$ (blue) and $N=128^3$ (dark blue). In all cases, data points correspond to the temporal averages, and contours denote intervals of one standard deviation.

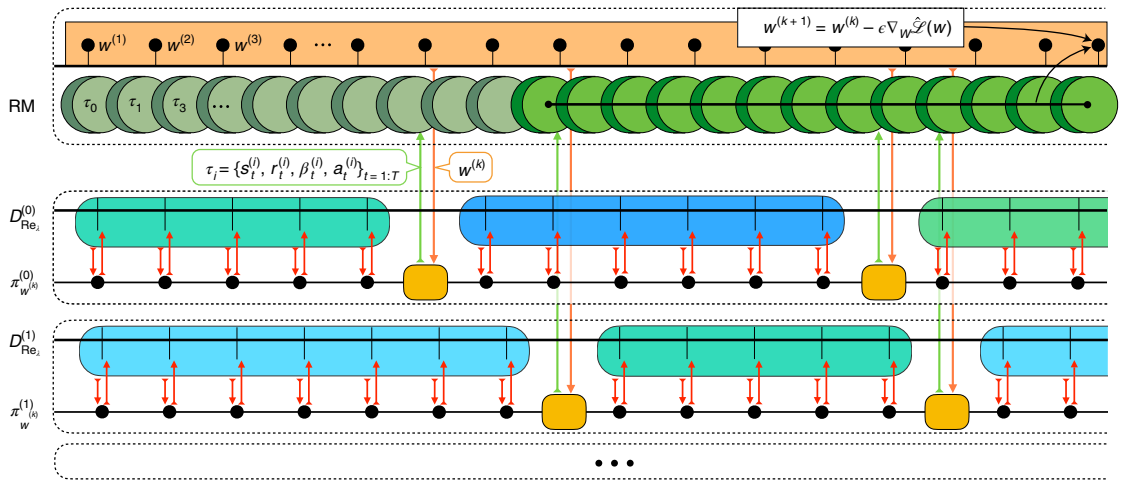

Fig. 5 | Schematic description of the training procedure implemented with the smarties library. Each dashed line represents a separate process. Multiple worker processes j run LES for randomly sampled $Re_\lambda \in \{65, 76, 88, 103, 120, 140, 163\}$. The Reynolds number determines the dynamics ($D^{(j)}_{Re_\lambda}$) and the number of RL steps per simulation (that is 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: (1) storing the N most recently collected RL steps into a RM and (2) 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 (that is one gradient step per SGS field update).

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 modelling 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 analyse the accuracy and generality of the trained model with respect to multiple quantities of interest.

New questions emerge from integrating deep learning and turbulence modelling. In the present study, the control policies trained by MARL (for example, π_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 transferable 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 modelling errors, a

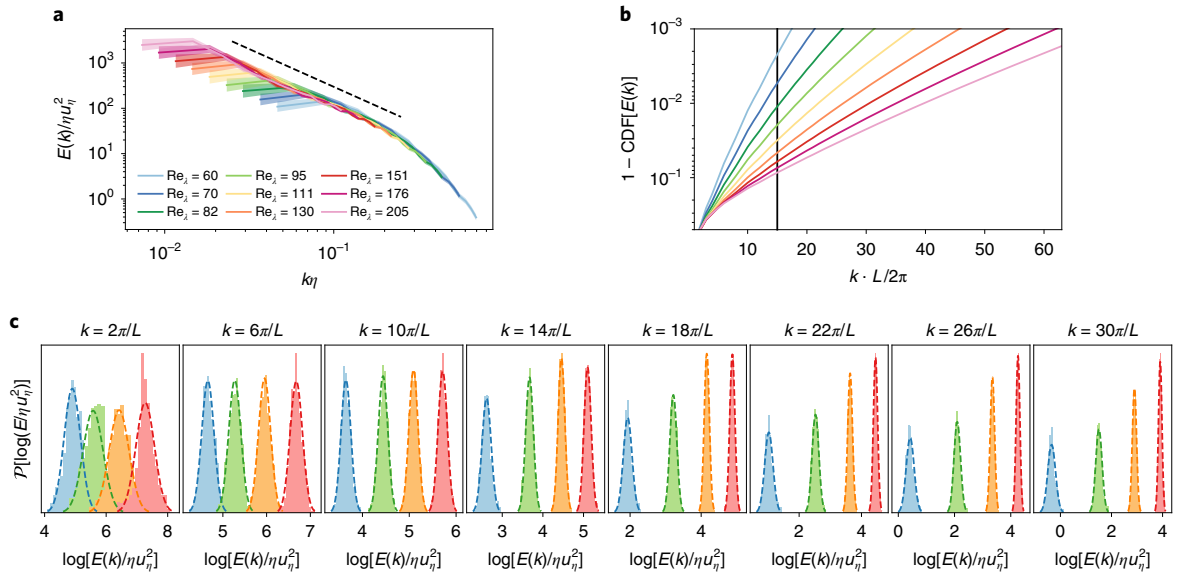

Fig. 6 | Statistical properties of DNS simulations of forced isotropic turbulence. a, Time-averaged energy spectra, and intervals of one standard deviation for log increments of $\mathrm{Re}_\lambda \in [60, 205]$ compared to Kolmogorov's spectrum $\propto k^{-5/3}$ (dashed line). **b**, Cumulative fraction of the kinetic energy 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 $\mathrm{Re}_\lambda = 65$ (blue), 88 (green), 110 (orange) and 163 (red) compared to a Gaussian fit.

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 modelling 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.

Methods

The RL 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. 2). The flow simulations are distributed across N_{workers} computational nodes ('workers'). The workers collect, for each agent, experiences organized into episodes:

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

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 'behaviour 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 an NN which, given input s_t, outputs the mean $\mu_w(s_t)$ and standard deviation $\sigma_w(s_t)$ of the policy π_w, and a state-value estimate $v_w(s_t)$. One gradient is defined per NN output. The statistics μ_w and σ_w are updated with the off-policy policy gradient (off-PG)[45]:

$$g^{\text{pol}}(w) = \mathbb{E}\Big[\big(\hat{q}_t - v_w(s_t)\big) \tfrac{\pi_w(a_t|s_t)}{\mathcal{P}(a_t|\mu_t,\sigma_t)} \nabla_w \log \pi_w(\mathrm{a_t}|\mathrm{s_t}) \Big| \{s_t, r_t, \mu_t, \sigma_t, a_t, \hat{q}_t\} \sim \mathrm{RM} \Big] \tag{4}$$

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]:

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

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 $\hat{q}_{T_{\text{end}}} \equiv 0$), and iteratively updated as individual steps are sampled. Retrace is also used to derive the gradient for the state-value estimate:

$$g^{\text{val}}(w) = \mathbb{E}\Big[\min\left\{1, \tfrac{\pi_w(a_t|s_t)}{\mathcal{P}(a_t|\mu_t,\sigma_t)}\right\}\big(\hat{q}_t - v_w(s_t)\big) \Big| \{s_t, r_t, \mu_t, \sigma_t, a_t, \hat{q}_t\} \sim \mathrm{RM} \Big] \tag{6}$$

The *off-PG* formalizes trial-and-error learning; it moves the policy to make actions with better-than-expected returns ($\hat{q}_t > v_w(s_t)$) more likely, and those with worse outcomes ($\hat{q}_t < v_w(s_t)$) less likely. Both equations (4) and (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 $\hat{g}(w) = \sum_{i=1}^{B} \hat{g}_i(w)$, 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 π_w, and the distribution of states that would be visited by π_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 $\pi_w(a_t|s_t)/\mathcal{P}(a_t|\mu_t,\sigma_t)$. 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:

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

Here $g_t^D(w) = \nabla_w D_{\mathrm{KL}}(\pi_w(\cdot|s_t) \;||\; \mathcal{P}(\cdot|\mu_t,\sigma_t))$ and $D_{\mathrm{KL}}(P \;||\; Q)$ 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 $\pi_w(a_t|s_t)$ 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:

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

Here $n_{\mathrm{far}}/N_{\mathrm{RM}}$ 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 set-up 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 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 (that is the RL objective of finding causal correlation between single actions and the observed reward) all the more difficult. The default values $\Delta t_{\mathrm{RL}} = \tau_\eta/8$ and $N_{\mathrm{agents}} = 4^3$ 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 $\mathrm{Re}_\lambda \in \{65, 76, 88, 103, 120, 140, 163\}$ 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 $T_{\mathrm{end}} = 20\tau_{\mathrm{l}}$ (between 750, if $\mathrm{Re}_\lambda = 65$, and 1,600, if $\mathrm{Re}_\lambda = 163$, actions per agent) or if $||\mathbf{u}||_\infty > 10^3 u_\eta$, which signals numerical instability. The policy πw 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 ref. [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 an RM of size $N_{\mathrm{RM}} = 10^6$. 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 $C = 1.5$ (equation (7)) and $D = 0.05$ (equation (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 $100\tau_l$.

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 equations,

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

where $S = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T)$ is the rate-of-strain tensor. The turbulent kinetic energy (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:

$$\frac{\mathrm{d}K}{\mathrm{d}t} = -\nu\int_{\mathcal{D}} ||\nabla\mathbf{u}||^2 + \int_{\mathcal{D}}\mathbf{uf} \; = -2\nu\langle Z\rangle \; + \; \langle\mathbf{u}\cdot\mathbf{f}\rangle \quad (11)$$

where the energy dissipation due to viscosity, is expressed in term of the norm of the vorticity $\omega \equiv \nabla\times\mathbf{u}$ 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 wavenumber 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:

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

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 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 $\epsilon_{\mathrm{tot}} = \epsilon_{visc} + \epsilon_{\mathrm{num}} + \epsilon_{\mathrm{sgs}}$ 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: $\eta = (\nu^3/\epsilon)^{1/4}$, $\tau_\eta = (\nu/\epsilon)^{1/2}$, $u_\eta = (\epsilon\nu)^{1/4}$. 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: $l_{\mathrm{I}} = \frac{3\pi}{4K}\int_0^\infty \frac{\tilde{E}(k)}{k}\mathrm{d}k$, $\tau_{\mathrm{I}} = \frac{l_{\mathrm{I}}}{\sqrt{2K/3}}$. The Taylor–Reynolds number is used to characterize flows with zero mean bulk velocity: $\mathrm{Re}_\lambda = K\sqrt{20/(3\nu\epsilon)}$.

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

Direct numerical simulations. 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 centred 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 $\mathrm{Re}_\lambda \in [60, 205]$ (Fig. 3a,b).

The initial velocity field is synthesized by generating a distribution of random Fourier coefficients matching a radial target spectrum $\tilde{E}(k)$ (ref. [51]): $\tilde{E}(k) = c_k\,\epsilon^{2/3}k^{-5/3}\,f_L(kL)\,f_\eta(k\eta)$, where $f_l(kl_{\mathrm{I}})$ 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. 3) are computed from 20 independent DNS with measurements taken every τ_η. Each DNS lasts $20\tau_{\mathrm{I}}$ and the initial $10\tau_{\mathrm{I}}$ are not included in the measurements, which found to be ample time to avoid the initial transient. Figure 3c shows that the distribution of energy content for each mode $\tilde{E}(k)$ is well approximated by a log-normal distribution such that $\log \mathrm{E}_{\mathrm{DNS}}^{\mathrm{Re}_\lambda} \sim \mathcal{N}\left(\mu_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}, \Sigma_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}\right)$, where $\mu_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}$ is the empirical average of the log-energy spectrum for a given Re_λ and $\Sigma_{\mathrm{DNS}}^{\mathrm{Re}_\lambda}$ is its covariance matrix.

Large-eddy simulations. LES[7] resolve the large-scale dynamics of turbulence and model their interaction with the SGS. The flow field $\overline{\mathbf{u}}$ on the grid is viewed as the result of filtering out the residual small-scales of a latent velocity field $\mathbf{u}$. The filtered Navier–Stokes equation for the field $\overline{\mathbf{u}}$ reads:

$$\frac{\partial\overline{\mathbf{u}}}{\partial t} + (\overline{\mathbf{u}}\cdot\nabla)\overline{\mathbf{u}} = -\nabla\overline{p} + \nabla\cdot\left(2\nu\overline{S} - \tau^R\right) + \overline{\mathbf{f}} \quad (14)$$

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

$$\tau^R = \overline{\mathbf{u}\otimes\mathbf{u}} - \overline{\mathbf{u}}\otimes\overline{\mathbf{u}}. \quad (15)$$

Closure equations are used to model the SGS motions represented by $\overline{\mathbf{u}\otimes\mathbf{u}}$.

The standard Smagorinsky model. The SSM[8] is a linear eddy-viscosity model that relates the residual-stress-tensor to the filtered rate of strain

$$\tau^R - \frac{1}{3}\mathrm{tr}(\tau^R) = -2\ \nu_t\ \overline{S} \tag{16}$$

$$\nu_t = (C_s\Delta)^2\ ||\overline{S}|| \tag{17}$$

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. Equation (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. The DSM[9] computes the parameter $C_s(\mathbf{x},t)$ 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:

$$\mathcal{L}_{\overline{\mathbf{u}}} = \overline{\mathbf{u}}\widehat{\otimes}\overline{\mathbf{u}} - \widehat{\overline{\mathbf{u}}}\otimes\widehat{\overline{\mathbf{u}}} = T^R - \widehat{\tau^R} \tag{18}$$

where $T^R = \widehat{\overline{\mathbf{u}}\otimes\overline{\mathbf{u}}} - \widehat{\overline{\mathbf{u}}}\otimes\widehat{\overline{\mathbf{u}}}$ 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 (equation (15)). If both residual stresses are approximated by a Smagorinsky model, the Germano identity becomes:

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

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

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

where $\mathcal{M} = ||\widehat{\overline{S}}||\widehat{\overline{S}} - (\widehat{\Delta}/\Delta)^2\ ||\widehat{\overline{S}}||\widehat{\overline{S}}$, 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 the total kinetic energy contained in the unresolved scales increases with Re_λ and decreases with the grid size (Fig. 3b). 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 higher Re_λ, the SGS model accounts for up to 10% of the total kinetic energy. 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 (ref. [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 $t = 100\tau_1$, disregarding the initial $10\tau_1$ 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 $r(\mathbf{x}^{(i)},t) = r^{\mathrm{grid}}(t) + r'(\mathbf{x}^{(i)},t)$. The base reward is a measure of the distance from the target DNS spectrum, derived from the regularized log-likelihood (equation (2)):

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

This regularized distance is preferred because a reward directly proportional to the probability $\mathcal{P}(\tilde{E}(t)|E_{\mathrm{DNS}}^{\mathrm{Re}_\lambda})$ (equation (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}:

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

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

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

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

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

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.

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

All the data analysed in this paper were produced with open-source software described in the code availability statement. Reference data and the scripts used to produce the data figures, as well as instructions to launch the reinforcement learning training and evaluate trained policies, are available on a GitHub repository (https://github.com/cselab/MARL_LES).

Code availability

Both direct numerical simulations and large-eddy simulations were performed with the flow solver CubismUP 3D (https://github.com/cselab/CubismUP_3D). The data-driven SGS models were trained with the reinforcement learning library smarties (https://github.com/cselab/smarties). The coupling between the two codes is also available through GitHub (https://github.com/cselab/MARL_LES).

Received: 15 May 2020; Accepted: 9 November 2020;
Published online: 04 January 2021

References

1. Kolmogorov, A. N. The local structure of turbulence in incompressible viscous fluid for very large reynolds numbers. *Dokl. Akad. Nauk SSSR* **30**, 299–301 (1941).
2. Taylor, G. I. Statistical theory of turbulence. Parts I and II. *Proc. R. Soc. Lon. A* **151**, 421–454 (1935).
3. Pope, S. B. *Turbulent Flows* (Cambridge Univ. Press, 2001).
4. Moin, P. & Mahesh, K. Direct numerical simulation: a tool in turbulence research. *Annu. Rev. Fluid Mech.* **30**, 539–578 (1998).
5. Moser, R. D., Kim, J. & Mansour, N. N. Direct numerical simulation of turbulent channel flow up to Re_τ= 590. *Phys. Fluids* **11**, 943–945 (1999).
6. Durbin, P. A. Some recent developments in turbulence closure modeling. *Annu. Rev. Fluid Mech.* **50**, 77–103 (2018).
7. Leonard, A. et al. Energy cascade in large-eddy simulations of turbulent fluid flows. *Adv. Geophys. A* **18**, 237–248 (1974).
8. Smagorinsky, J. General circulation experiments with the primitive equations: I. The basic experiment. *Mon. Weather Rev.* **91**, 99–164 (1963).
9. Germano, M., Piomelli, U., Moin, P. & Cabot, W. H. A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids A* **3**, 1760–1765 (1991).
10. Lilly, D. K. A proposed modification of the germano subgrid-scale closure method. *Phys. Fluids A* **4**, 633–635 (1992).
11. Lee, C., Kim, J., Babcock, D. & Goodman, R. Application of neural networks to turbulence control for drag reduction. *Phys. Fluids* **9**, 1740–1747 (1997).
12. Milano, M. & Koumoutsakos, P. Neural network modeling for near wall turbulent flow. *J. Comput. Phys.* **182**, 1–26 (2002).
13. Duraisamy, K., Iaccarino, G. & Xiao, H. Turbulence modeling in the age of data. *Annu. Rev. Fluid Mech.* **51**, 357–377 (2019).
14. Sarghini, F., De Felice, G. & Santini, S. Neural networks based subgrid scale modeling in large eddy simulations. *Comput. Fluids* **32**, 97–108 (2003).
15. Gamahara, M. & Hattori, Y. Searching for turbulence models by artificial neural network. *Phys. Rev. Fluids* **2**, 054604 (2017).
16. Xie, C., Wang, J., Li, H., Wan, M. & Chen, S. Artificial neural network mixed model for large eddy simulation of compressible isotropic turbulence. *Phys. Fluids* **31**, 085112 (2019).
17. Vollant, A., Balarac, G. & Corre, C. Subgrid-scale scalar flux modelling based on optimal estimation theory and machine-learning procedures. *J. Turbul.* **18**, 854–878 (2017).
18. Hickel, S., Franz, S., Adams, N. & Koumoutsakos, P. Optimization of an implicit subgrid-scale model for LES. In *Proc. 21st International Congress of Theoretical and Applied Mechanics* (Springer, 2004).
19. Maulik, R. & San, O. A neural network approach for the blind deconvolution of turbulent flows. *J. Fluid Mech.* **831**, 151–181 (2017).
20. Sirignano, J., MacArt, J. F. & Freund, J. B. DPM: A deep learning PDE augmentation method with application to large-eddy simulation. *J. Comput. Phys.* **423**, 109811 (2020).
21. Wu, J.-L., Xiao, H. & Paterson, E. Physics-informed machine learning approach for augmenting turbulence models: a comprehensive framework. *Phys. Rev. Fluids* **3**, 074602 (2018).
22. Nadiga, B. & Livescu, D. Instability of the perfect subgrid model in implicit-filtering large eddy simulation of geostrophic turbulence. *Phys. Rev. E* **75**, 046303 (2007).
23. Beck, A., Flad, D. & Munz, C.-D. Deep neural networks for data-driven LES closure models. *J. Comput. Phys.* **398**, 108910 (2019).

24. Sutton, R. S. & Barto, A. G. *Reinforcement Learning: An Introduction* 2nd edn (MIT Press, 2018).
25. Mnih, V. et al. Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015).
26. Silver, D. et al. Mastering the game of go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
27. Levine, S., Finn, C., Darrell, T. & Abbeel, P. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **17**, 1334–1373 (2016).
28. Akkaya, I. et al. Solving Rubik's cube with a robot hand. Preprint at https://arxiv.org/abs/1910.07113 (2019).
29. Garnier, P. et al. A review on deep reinforcement learning for fluid mechanics. Preprint at https://arxiv.org/abs/1908.04127 (2019).
30. Gazzola, M., Hejazialhosseini, B. & Koumoutsakos, P. Reinforcement learning and wavelet adapted vortex methods for simulations of self-propelled swimmers. *SIAM J. Sci. Comput.* **36**, B622–B639 (2014).
31. Reddy, G., Celani, A., Sejnowski, T. J. & Vergassola, M. Learning to soar in turbulent environments. *Proc. Natl Acad. Sci. USA* **113**, E4877–E4884 (2016).
32. Novati, G. et al. Synchronisation through learning for two self-propelled swimmers. *Bioinspir. Biomim.* **12**, 036001 (2017).
33. Verma, S., Novati, G. & Koumoutsakos, P. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proc. Natl Acad. Sci. USA* **115**, 5849–5854 (2018).
34. Belus, V. 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. Biferale, L., Bonaccorso, F., Buzzicotti, M., Clark Di Leoni, P. & Gustavsson, K. Zermelo's problem: optimal point-to-point navigation in 2D turbulent flows using reinforcement learning. *Chaos* **29**, 103138 (2019).
36. Novati, G., Mahadevan, L. & Koumoutsakos, P. Controlled gliding and perching through deep-reinforcement-learning. *Phys. Rev. Fluids* **4**, 093902 (2019).
37. François-Lavet, V. et al. *An Introduction to Deep Reinforcement Learning* 219–354 (Foundations and Trends in Machine Learning Vol. 11, 2018).
38. Ling, J., Kurzawski, A. & Templeton, J. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166 (2016).
39. Pope, S. A more general effective-viscosity hypothesis. *J. Fluid Mech.* **72**, 331–340 (1975).
40. Buşoniu, L., Babuška, R. & De Schutter, B. in *Innovations in Multi-Agent Systems and Applications – 1* (eds Srinivasan, D. & Jain, L. C.) 183–221 (Springer, 2010).
41. Novati, G. & Koumoutsakos, P. Remember and forget for experience replay. In *Proc. 36th International Conference on Machine Learning* **97**, 4851–4860 (2019).
42. Lin, L. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach. Learn.* **8**, 69–97 (1992).
43. Meyers, J., Geurts, B. J. & Baelmans, M. Database analysis of errors in large-eddy simulation. *Phys. Fluids* **15**, 2740–2755 (2003).
44. Zhiyin, Y. Large-eddy simulation: past, present and the future. *Chin. J. Aeronaut.* **28**, 11–24 (2015).
45. Degris, T., White, M. & Sutton, R. S. Off-policy actor-critic. In *Proc. 29th International Conference on Machine Learning* 179–186 (2012).
46. Munos, R., Stepleton, T., Harutyunyan, A. & Bellemare, M. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems 29* 1054–1062 (2016).
47. Glorot, X. & Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proc. 13th International Conference on Artificial Intelligence and Statistics* **9**, 249–256 (2010).
48. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. In *Proc. 3rd International Conference on Learning Representations (ICLR)* (2014).
49. Ghosal, S., Lund, T. S., Moin, P. & Akselvoll, K. A dynamic localization model for large-eddy simulation of turbulent flows. *J. Fluid Mech.* **286**, 229–255 (1995).
50. Chorin, A. J. A numerical method for solving incompressible viscous flow problems. *J. Comput. Phys.* **2**, 12–26 (1967).
51. Rogallo, R. S. & Moin, P. Numerical simulation of turbulent flows. *Annu. Rev. Fluid Mech.* **16**, 99–137 (1984).

Acknowledgements

We are very grateful to H. J. Bae (Harvard University) and A. Leonard (Caltech) for insightful feedback on the manuscript, and to J. Canton and M. Boden (ETH Zürich) for valuable discussions throughout the course of this work. We acknowledge support by the European Research Council Advanced Investigator Award 341117. Computational resources were provided by the Swiss National Supercomputing Centre (CSCS) Project s929.

Author contributions

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

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s42256-020-00272-0.

Correspondence and requests for materials should be addressed to P.K.

Peer review information *Nature Machine Intelligence* thanks Elie Hachem, Jonathan Freund and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

nature research

Corresponding author(s): Petros Koumoutsakos

Last updated by author(s): Jun 8, 2020

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	All data was produced with open-source software. Flow simulations were performed with the flow solver CubismUP 3D (https://github.com/cselab/CubismUP_3D). The data-driven SGS models were trained with the Reinforcement Learning library smarties (https://github.com/cselab/smarties).
Data analysis	In a sub-directory of the repository of smarties (https://github.com/cselab/smarties/tree/master/apps/CUP3D_LES_HIT) we provide the python scripts used to produce the figures of this paper (which rely on well established numpy and scipy routines).

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

Reference data and a pre-trained RL model is provided in a sub-directory of the repository of smarties (https://github.com/cselab/smarties/tree/master/apps/CUP3D_LES_HIT).

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☐ Life sciences ☐ Behavioural & social sciences ☒ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Ecological, evolutionary & environmental sciences study design

All studies must disclose on these points even when the disclosure is negative.

Study description	The paper proposes applying reinforcement learning to turbulence modeling for large-eddy simulations (LES). The merits of the proposed methods are validated by performing turbulent flow LES with both established and data-driven sub-grid-scale closures. The time averaged statistical properties (and distributions via standard deviations) are compared to reference fully-resolved simulations which are orders of magnitude more expensive, and unfeasible for many engineering applications.
Research sample	Because the LES have limited computational cost, we are able to analyze their statistical properties by time integration for long time horizons, details are reported in the Supplementary Information.
Sampling strategy	The time horizons over which data is sampled are one order or magnitude longer than the time required to reach statistical steady state. Acquisition of additional data was found to have no effect on the reported statistics.
Data collection	Data is collected by conducting flow simulations coupled with the RL-driven control strategy.
Timing and spatial scale	Data is acquired by post-processing flow snapshots of the entire flow at uniform time intervals.
Data exclusions	No data were excluded.
Reproducibility	We obtained quantitatively consistent accuracy measurements by multiple, independently seeded, trained RL policies.
Randomization	Turbulent flow is chaotic. The statistical properties of the prototypical turbulent flow considered in the paper are fully defined by one non-dimensional number (the Reynolds number). Therefore, there are no covariates.
Blinding	Blinding is not applicable to flow simulations.

Did the study involve field work? ☐ Yes ☒ No

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☒	☐ Human research participants
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging