AI super-resolution subfilter modeling for multi-physics flows

Mathis Bode
m.bode@fz-juelich.de
Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich GmbH
Jülich, NRW, Germany

Figure 1: Visualization of the interface of a gas/liquid flow.

ABSTRACT

Many complex simulations are extremely expensive and hardly if at all doable, even on current supercomputers. A typical reason for this are coupled length and time scales in the application which need to be resolved simultaneously. As a result, many simulation approaches rely on scale-splitting, where only the larger scales are simulated, while the small scales are modeled with subfilter models. This work presents a novel subfilter modeling approach based on AI super-resolution. A physics-informed enhanced super-resolution generative adversarial network (PIESRGAN) is used to accurately close subfilter terms in the solved transport equations. It is demonstrated how a simulation design with the PIESRGAN-approach can be used to accelerate complex simulations on current supercomputers, on the example of three fluid dynamics simulation setups with complex features on the supercomputer environment JURECA-DC/JUWELS (Booster). Further advantages and shortcoming of the PIESRGAN-approach are discussed.

CCS CONCEPTS

• **Computing methodologies** → *Massively parallel and high-performance simulations.*

KEYWORDS

AI Super-Resolution, Large-Eddy Simulation, Fluid Mechanics, Combustion, Multiphase, Turbulence

PASC '23, June 26–28, 2023, Davos, Switzerland

ACM ISBN 979-8-4007-0190-0/23/06.
https://doi.org/10.1145/3592979.3593414

ACM Reference Format:
Mathis Bode. 2023. AI super-resolution subfilter modeling for multi-physics flows. In *Platform for Advanced Scientific Computing Conference (PASC '23), June 26–28, 2023, Davos, Switzerland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3592979.3593414

1 INTRODUCTION

Computing flow problems by solving the Navier-Stokes equations or equations derived from them, potentially coupled with multi-physics phenomena, belongs to the most established high-performance computing (HPC) applications, yet still some problems remain impossible to solve even using today's fastest supercomputing systems. Typical engineering-type flow applications have in common that a simulation needs to properly discretize the large-scale flow behavior, such as enforced by geometry features. Turbulent flows, however, come with an additional challenge: All length scales are coupled, requiring to fully resolve all scales down to the smallest scale of fluid motion to accurately predict its behavior. Additionally, the more turbulent a flow is, measured by the Reynolds number, i. e., the ratio of inertial forces to viscous forces, the wider large and small scales are separated [45]. Therefore, direct numerical simulations (DNSs), which aim to resolve all scales, become very expensive for common engineering problems, which are often strongly turbulent to, e. g., benefit from faster mixing time scales. Thus, many flow simulations require modeling of smaller scales to become computationally affordable. One commonly used modeling approach is large-eddy simulation (LES) [44, 45]. In LES, the flow scales are decoupled by a filter operation, solving only for the larger scales. The effect of the small scales below the filter on the larger scales is modeled by means of subfilter models, such as the Smagorinsky model [47].

A model transport equation for a scalar ϕ over time t reads

$$\partial_t \left(\rho\phi\right) + \nabla \cdot \left(\rho\phi\mathbf{u}\right) = \nabla \cdot \left(D\nabla \left(\rho\phi\right)\right) + \dot{\omega}, \tag{1}$$

where ∇ is the del operator corresponding to spatial coordinates, ρ the density, $\mathbf{u}$ the velocity vector, D the molecular diffusivity, and $\dot{\omega}$ the source term. Solving (1) directly along with other equations, such as for density, momentum, and energy, would be called DNS if the simulation resolves all relevant scales sufficiently. A filter kernel $G(\mathbf{r})$ is used to decouple the large and small scales in LES. This operation can be defined as

$$\overline{\{\cdot\}}(\mathbf{x}) = \iiint G(\mathbf{r})\{\cdot\}(\mathbf{x}-\mathbf{r})\,\mathrm{d}\mathbf{r}, \tag{2}$$

where an overbar denotes filtered quantities. Examples for G are the symmetric Gaussian filter kernel (cf. Fig. 3) and wavenumber cut-off filter kernels. For compressible and variable density flows, it is convenient to work with Favre-filtering, denoted with a tilde and defined as $\overline{\rho}\widetilde{\{\cdot\}} = \overline{\rho\{\cdot\}}$. A quantity can then be split into a filtered part and the subfilter contribution as $\{\cdot\} = \widetilde{\{\cdot\}} + \{\cdot\}''$ with $\widetilde{\{\cdot\}}'' = 0$. Applied to (1) and assuming constant diffusivity and Reynolds operators, the Favre-filtered equation reads

$$\partial_t \left(\overline{\rho}\widetilde{\phi}\right) + \nabla \cdot \left(\overline{\rho}\widetilde{\phi}\widetilde{\mathbf{u}}\right) = D\nabla^2 \left(\overline{\rho}\widetilde{\phi}\right) + \overline{\dot{\omega}} - \nabla \cdot \left(\overline{\rho}\widetilde{\phi''\mathbf{u}''}\right), \tag{3}$$

which looks very similar to the unfiltered equation (1) but features one additional term, put as last term here. All but this last term contain only filtered quantities, which can be fully resolved, as the very small scales, which might be impossible to resolve, are removed by the filter. However, the additional term in (3) contains subfilter contributions, which cannot be resolved in LES and thus this term is unclosed and requires modeling.

Turbulent combustion is a particular challenging engineering flow problem, which requires to accurately predict chemistry. This can be done by low-order combustion models, such as flamelet approaches [6, 15], or by solving a set of coupled partial differential equations (PDEs) for individual species along with the flow equations, often referred to as finite-rate-chemistry model. While low-order combustion models are sufficient for multiple engineering problems, they are currently not able to predict all different flame-related phenomena accurately and thus are a topic of ongoing research. Finite-rate-chemistry models on the other hand are able to consider most flame-effects as long as a sufficient reaction mechanism can be used but make the simulations much more expensive than turbulent-only cases and the resulting data are very big, as the data of all scalar fields need to be stored. Furthermore, the smallest chemistry-related scales can be even smaller than the flow scales and thus require an even finer mesh to, e. g., sufficiently discretize a reaction zone [44]. Overall, simulations of turbulent flows with combustion quickly become prohibitively expense and are thus often run on the largest supercomputers available. Furthermore, an accurate prediction of emissions with reduced order models is still challenging, as emphasized by recent literature [14, 19, 29, 37, 51], and thus, a turbulent premixed combustion problem makes an excellent target case for the present work.

Multiphase flows are another important problem class, and particularly interfacial flows [50] are tough. During breakup or ligament formation, infinitesimally small structures appear, which are typically impossible to fully resolve. Therefore, simulations often become grid dependent. As the accurate prediction of interfaces is often the first step for predicting subsequent processes, such as chemistry that occurs at the interface, improved predictability with reduced order models or in LES is crucial and the second target case presented here.

This paper presents the application of AI super-resolution-based subfilter models for LES, which have been recently developed [1–5, 11–13], and focuses on their prediction quality for multi-physics problems. AI super-resolution or single image super-resolution (SISR) problems have gained a lot of attention from the computer science community. In such SISR problems, machine learning (ML) or deep learning (DL) techniques are employed to add information into images to increase the image resolution (i. e., to super-resolve the image). Usually, a network is trained with a large number of images to extract and learn features which are consecutively added to the super-resolved target image based on local information. Thus, they exceed classical techniques, such as bicubic interpolation. Reference [17] introduced a super-resolution convolutional neural network (SRCNN), a deep convolutional neural network (CNN) for directly learning the end-to-end mapping between images with low and high resolution. Their approach has been continuously improved [18, 31, 33, 34, 36, 46, 49, 53] to correct multiple shortcomings, such as oversmoothed results, and improve the prediction accuracy by, e. g., introducing the concept of perceptual loss in order to better predict high frequency details. Reference [40] suggested to use generative adversarial networks (GANs) [28] instead of CNNs, which were further updated to the enhanced super-resolution GAN (ESRGAN) [52].

The ambition to add information based on the local state makes AI super-resolution a promising tool for any kind of underresolved simulation and experimental data, such as from satellites for weather prediction or many HPC problems, ranging from climate research

[48] to cosmology [41]. However, this work focuses on the application of AI super-resolution for LES subfilter modeling to efficiently advance flow simulations in time. For that, a decaying turbulence case is discussed as base case. Afterward, two relevant engineering problems, a turbulent premixed flame kernel setup and a temporally evolving interfacial jet, are used to emphasize the prediction accuracy of PIESRGAN for complex multi-physics flows. Physical aspects of the three demonstration cases are not in the scope of this work and hence are only briefly discussed, as needed in order to emphasize the advantages of PIESRGAN.

LES is a widely established approach in fluid dynamics. However, until today, no general closure model exists and reproduction of complex flows with LES is still a challenge. Reference [13] introduced PIESRGAN as systematic subfilter framework for LES and showed accurate results for turbulence and a simple combustion case. However, they explicitly state that the computing performance is still a big issue, preventing PIESRGAN-subfilter models from application in relevant LESs. Therefore, the focus of this work is to describe the usage of PIESRGAN on current supercomputers and demonstrate how it can be used in practically relevant cases. The three major contributions of this work are:

- An efficient implementation of PIESRGAN for HPC systems is described and discussed.
- The prediction quality for multiple multi-physics is given, and generality aspects are emphasized.
- Industrially relevant workflows are discussed and outlined.

This paper is structured as follows: The next section explains the AI super-resolution subfilter modeling approach, including the architecture, the algorithm, and implementation details. Afterwards, three HPC simulations are used to demonstrate the prediction accuracy of the AI super-resolution-based modeling approach. Industrially relevant and generality aspects are discussed. The paper finishes with conclusions.

2 AI SUPER-RESOLUTION SUBFILTER MODELING

This section describes the employed architecture for all examples in this work, the algorithm for advancing simulations with AI super-resolution-based subfilter modeling, and gives implementation details. In more detail, the so-called physics-informed enhanced super-resolution GAN (PIESRGAN) [13] is used, which extended and modified ESRGAN for flow LES-subfilter modeling and follows a hybrid approach. The subfilter modeling is done by PIESRGAN, but the simulation as a whole is advanced classically with the filtered equations, i. e., the time integration is not incorporated within the neural network, as, e. g., done in [26]. An important advantage of keeping time integration and subfilter modeling separated is the universality with respect to, e. g., different geometries and setups. Furthermore, the equations for advancing in time are well-known and a lot of experience exists. In PIESRGAN, physically motivated conditions are enforced as part of the loss function, i. e., the target function, which is minimized at training of the network.

To elaborate further, in many engineering problems, the resulting flow topologies often depend on the chosen boundary conditions enforced by the geometry or setup. Therefore, it is often desirable to rely on universality at the smallest scales and to compute the topology by solving the equations in time. This may be different for other applications, such as Earth system modeling. There, the boundary conditions are typically very similar, and thus the advantages of including time integration in the network may prevail. In general, data-driven methods work best as long as the problems have some universal behavior that can be learned. Consequently, the data-driven approach should preferably work on this scale of universality, which depends on the application.

The subfilter modeling approach technically works with any kind of data. However, it makes most sense if the data feature some kind of universality, which can be learned. As an example, turbulence is known to feature some universality on the smallest scales [24]. Therefore, it is an ideal application for any kind of super-resolution. AI super-resolution also works for quantities without universal behavior on the smallest scales, as, e. g., reacting chemical species. However, it was found that it requires to solve additional equations on the reconstructed fields to improve the accuracy [1].

2.1 Architecture

PIESRGAN is a GAN model, which are generative models that aim to estimate the unknown probability density of observed data without an explicitly provided data likelihood function, i. e., with unsupervised learning. Technically, a GAN has two networks: The generator network is used for modeling and creates new modeled data. The discriminator tries to distinguish whether data are generator-created or real data and provides feedback to the generator network. Thus, as the learning process advances, the generator gets better at creating data as close as possible to real data, and the discriminator learns to better identify fake data. This process can be thought of as two players carrying out a minimax zero-sum game to estimate the unknown data probability distribution.

The network architecture and training process are sketched in Fig. 2. Fully resolved 3-dimensional (3-D) data ("H") are filtered to get filtered data ("F"). The filtered data is used as input to the generator for creating the reconstructed data ("R"). The accuracy of the reconstructed data is evaluated by means of the fully resolved data. The discriminator tries to distinguish between reconstructed and fully resolved data. The accuracy of the reconstruction is measured by means of the loss function, which reads

$$\mathcal{L} = \beta_1 L_{\text{adversarial}} + \beta_2 L_{\text{pixel}} + \beta_3 L_{\text{gradient}} + \beta_4 L_{\text{physics}}, \quad (4)$$

where β_1, β_2, β_3, and β_4 are coefficients weighting the different loss term contributions, with $\sum_i \beta_i = 1$. In (4), the adversarial loss is the discriminator/generator relativistic adversarial loss [32], which measures both how well the generator is able to create accurate reconstructed data compared to the fully resolved data and how well the discriminator is able to identify fake data. The pixel loss and the gradient loss are defined using the mean-squared error (MSE) of the quantity itself and of the gradient of the quantity, respectively. The physics loss enforces physically motivated conditions, such as the conservation of mass, species, and elements, depending on the underlying physics of the problem. For the premixed flame kernel application in this work, it reads

$$L_{\text{physics}} = \beta_{41} L_{\text{mass}} + \beta_{42} L_{\text{species}}, \quad (5)$$

where β_{41} and β_{42} are coefficients weighting the different physical loss term contributions, with $\sum_i \beta_{4i} = 1$. Element conservation was

found to be not important for this case, and therefore, the potential term $\beta_{43} L_{\text{elements}}$ in (5) was set to zero here, i. e., $\beta_{43} = 0$, and is omitted. However, the importance of this term depends on the chemical mechanism used and is usually not known before testing [1, 3, 11]. If numerical tests show that this term can be omitted, this simplifies the training process. The physically motivated loss term is very important for the application of PIESRGAN to flow problems. If the conservation laws are not very well fulfilled, the simulations tend to blow up rapidly, which is an important difference to super-resolution in the context of images. Errors, which might be acceptable there, can be easily too large for the usage as subfilter model [13].

The generator heavily uses 3-D CNN layers (Conv3D) [35] with kernel size of 3 and stride 1 combined with leaky rectified linear unit (LeakyReLU) layers for activation [42]. The residual in residual dense block (RRDB), which was introduced for ESRGAN is essential for the performance of state-of-the-art super-resolution. It replaced the residual block (RB) employed in previous architectures and contains fundamental architectural elements such as residual dense blocks (RDBs) with skip-connections. A residual scaling factor β_{RSF} helps to avoid instabilities in the forward and backward propagation. RDBs use dense connections inside. The output from each layer within the dense block (DB) is sent to all the following layers. The discriminator network is simpler. It inherits basic CNN layers (Conv3D) combined with LeakyReLU layers for activation with and without batch normalization (BN). The final layers contain a fully connected layer with LeakyReLU and dropout with dropout factor β_{dropout}. For all cases in this work, 80 layers for the generator network and 28 layers for the discriminator network were used. A summary of all hyperparameters is given in Tab. 1. The hyperparameter value range was evaluated by a hyperparameter study with multiple physical applications, partly supported by AutoML. Generally, the network hyperparameters were found to be robust with respect to the different test cases, i. e., a working combination for one case usually results also in sufficiently accurate results for other cases. However, the complex combustion cases presented in this work require more physically motivated terms as part of the loss function than the cases discussed in [13]. As the additional terms for L_{physics} implicitly lower the weight for the originally mass conversation enforcement term, the range for β_4 was increased. Consecutively, the lower bound for β_2 was reduced to fulfill the summation condition.

Table 1: Overview of the PIESRGAN hyperparameters. The given ranges represent the sensitivity intervals with acceptable network results. The central values were used for the decaying turbulent case in this work.

β_1	$[0.2 \times 10^{-5}, 0.6 \times 10^{-4}, 0.8 \times 10^{-4}]$
β_2	$[0.721, 0.890, 0.918]$
β_3	$[0.04, 0.06, 0.15]$
β_4	$[0.01, 0.05, 0.18]$
β_{RSF}	$[0.1, 0.2, 0.3]$
β_{dropout}	$[0.2, 0.4, 0.5]$
$l_{\text{generator}}$	$[1.2 \times 10^{-6}, 4.5 \times 10^{-6}, 5.0 \times 10^{-6}]$
$l_{\text{discriminator}}$	$[4.4 \times 10^{-6}, 4.5 \times 10^{-6}, 8.5 \times 10^{-6}]$

2.2 Algorithm

The established LES equations are used to advance a PIESRGAN-LES in time. As a consequence of the filter operation to the equations, unclosed terms appear, which require information from below the filter width to be evaluated. The LES subfilter algorithm aims to reconstruct this required information to close the LES equations. This is done during every time step. For the cases including chemistry, the chemistry can be included in the PIESRGAN during the training process [1]. As chemistry is often only locally active, this can be also used to save computing time by adaptively solving only in relevant regions. The algorithm starts with the LES solution Φ_{F}^{n} at time step n, which includes the entirety of all relevant fields in the simulation, and consists of repeating the following steps:

(1) Use the PIESRGAN to reconstruct Φ_{R}^{n} from Φ_{LES}^{n}.
(2) (Only for nonuniversal quantities) Use Φ_{R}^{n} to update the scalar fields of Φ to $\Phi_{\text{R}}^{n;\text{update}}$ by solving the unfiltered scalar equations on the mesh of Φ_{R}^{n}.
(3) Use $\Phi_{\text{R}}^{n;\text{update}}$ to estimate the unclosed terms Ψ_{LES}^{n} in the LES equations of Φ for all fields by evaluating the local terms with $\Phi_{\text{R}}^{n;\text{update}}$ and applying a filter operator.
(4) Use Ψ_{LES}^{n} and Φ_{LES}^{n} to advance the LES equations of Φ to Φ_{LES}^{n+1}.

2.3 Training Details

For all shown examples, the data were split in data for training and testing to avoid reproduction of fully seen data. During the training and querying processes, it was found that consistent normalization of quantities is very important for highly accurate results [13]. For example, turbulence and combustion feature some quantities which show a logarithmic behavior. A normalization and transformation to a nonlogarithmic scale supports the learning capability of the neural network. Furthermore, the training and the reconstruction are done based on subboxes, as reconstructing too big boxes at once can become very memory intensive. Typically, each subbox is chosen large enough to cover the relevant physical scales [13], and sizes of 16^3 to 32^3 gave good results for the test cases presented in this work. The filter width can become problematic if nonuniform meshes are employed. In these cases, training with multiple filter widths is suggested to achieve good accuracy throughout the full domain [3].

One usual challenge during the training process is the initialization of the network weights. To simplify the training process, the trained weights of the turbulence-only case were used to initialize the weights of the networks for the combustion cases. Afterwards generator and discriminator can be further updated with the case-specific data. However, it was found that a further update of the discriminator weights is often not desirable, and generator-only updates lead to better general prediction results.

The potential extrapolation capabilities of data-driven methods is always an issue. Many trained networks only work well in regions which were accessible during the training process. This can become very problematic for flow applications, where often data at low Reynolds numbers is abundant, while data at high Reynolds numbers is not computable at all, making transfer learning difficult. To deal with this problem, concepts, such as two-step training

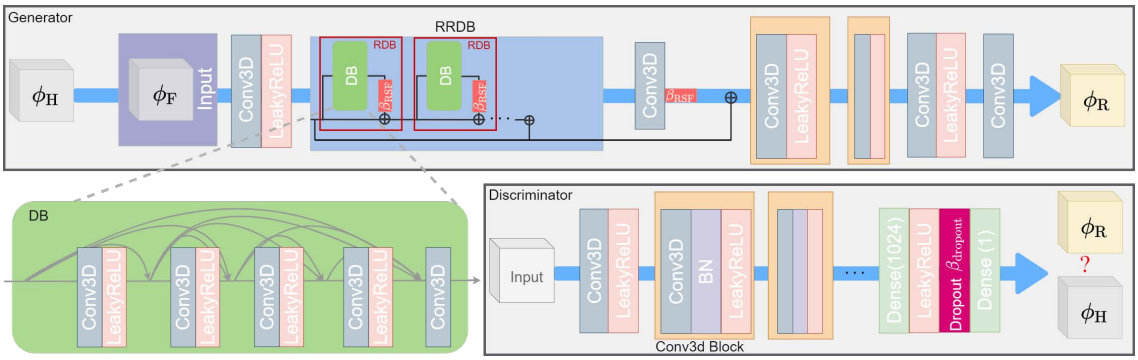

Figure 2: Sketch of PIESRGAN. "H" denotes high-fidelity data, such as DNS data, "F" are corresponding filtered data, and "R" are the reconstructed data. The components are: Conv3D - 3D Convolutional Layer, LeakyReLU - Activation Function, DB - Dense Block, RDB - Residual Dense Block, RRDB - Residual in Residual Dense Block, β_{RSF} - Residual Scaling Factor, BN - Batch Normalization, Dense - Fully Connected Layer, Dropout - Regularization Component, β_{dropout} - Dropout Factor. Image from [13] which is under CC BY 4.0 license.

approaches [13], can be used, relying on the enhanced prediction capabilities of GANs, as compared to single networks [1]. This open question is outside of the scope of the present work, hence only interpolation cases are discussed here.

2.4 High-Performance Computing Implementation Details

State-of-the-art supercomputers get most of their performance from GPUs. As an example, each JURECA-DC GPU cluster node at Jülich Supercomputing Centre, Forschungszentrum Jülich, features two AMD EPYC 7742 CPUs with a total of 128 cores and four Nvidia A100 GPUs resulting in 2322 EFLOP. 94.4 % of this performance comes from the GPUs, while all CPU cores only account for 5.6 %. Any efficient HPC implementation must therefore make maximum use of the GPUs, and to efficiently employ GPUs, PIESRGAN was implemented using a TensorFlor/Keras framework with OpenMP, MPI, and CUDA. The implementation details for training and simulation with PIESRGAN are described in the following two sections.

The implementation could be simplified if the computing power of the CPUs would be neglected and everything would be computed on the GPUs. Considering the FLOP performance, this would only lead to small losses on modern computing nodes, as discussed for the JURECA-DC GPU cluster nodes. However, two practical reasons oppose: First, for many cluster nodes, the ratio between CPU cores and number of GPUs is larger, shifting the available computing power in the direction of the CPUs. Second, and more importantly, many complex compute codes for simulations are not ported to GPUs yet and might never be. Therefore, they would not be able to employ a GPU-only approach, making a high portability unavoidable considering currently used software frameworks. As a consequence, the approach pursued in this work relies on a well-defined API, as described for the simulation workflow below. However, using CPUs and GPUs also means that an imbalance between load on CPUs and GPUs might thwart the overall simulation performance, as, e. g., the GPUs often need to wait until the CPUs finished advancing their equations. As mentioned before, the size of the reconstructed subboxes can be used to balance this load. The bigger the subboxes, the more expensive for the GPUs, but the less number of cells are required on the LES mesh processed by the CPUs. Note that the API can be also used with CPUs on both sides, enabling to run PIESRGAN-LES on CPU-only clusters as well.

2.4.1 Training Workflow. Two different training situations need to be distinguished: Training with stored data and on-the-fly training during simulations without explicitly storing the data. The employed implementation with stored data is trivial. A outer loop implemented in Python loads the data and generates data chunks which are then distributed to GPUs for training. Filtering and other preprocessing steps are done separately. The on-the-fly training is more advanced. For that, a hybrid parallelization with OpenMP (on node level) and MPI (between nodes) is employed. The advantage of this implementation is that the CPU cores per node can be distributed in cores used for the simulation and cores used for gathering and preparing the current results for training on the GPUs, which are accessed with CUDA. Due to the OpenMP parallelization on node level, the cores for data processing have access to all data. Filtering of the data is done on the GPUs. On-the-fly training was used for the premixed combustion case in this work. A ratio of four data CPU cores to 28 simulation CPU cores resulted in an efficient execution. On-the-fly training significantly reduces the cost for data movement and storing, however, it has the limitation that only one training iteration is possible.

2.4.2 Simulation Workflow. To perform PIESRGAN-LES at full scale of current supercomputers, an efficient HPC implementation is required. The implementation of PIESRGAN utilized in this work is fully executed on GPUs and connected to the CPUs, which advance the flow equations, by an API. The application of the filtering as part of PIESRGAN on the GPUs is crucial for two reasons: First, the filter operation is not cheap, and the execution as tensor operation on the GPUs is time-efficient. Second, the filtering significantly reduces the amount of data transfer between CPUs and GPUs, as data discretized on the coarser LES mesh are transferred in both directions. The PIESRGAN simulation workflow was implemented in CIAO for this work, which employs finite differences to solve

the Navier-Stokes equations along with multi-physical effects and chemistry and has been used for many DNS and LES studies in recent years [6, 7, 9, 10, 16, 23].

To facilitate the reproducibility of this work and clarify more technical details, a basic version of PIESRGAN is available on GitLab (https://git.rwth-aachen.de/Mathis.Bode/PIESRGAN.git). Furthermore, PIESRGAN is available as software as a service (SaaS) in the supercomputing environment at the Jülich Supercomputing Centre as JUelich Large-Eddy Simulation (JuLES), which has just been awarded as one winner of RWTH Aachen University's Innovation Award 2022.

3 RESULTS AND DISCUSSION

The accuracy is a key measurement for any new modeling approach. Therefore, it is discussed for three different applications in this section: a decaying turbulence case, a turbulent premixed flame kernel setup, and an interfacial temporal jet simulation. All simulations were performed using world-leading HPC resources, consuming altogether over 100×10^6 core-h. The discussion gives *a priori*, i. e., the model employed within one time step without feedback to the simulation, and *a posteriori*, i. e., the continuous application of the model to advance the simulation in time using the model results, evaluation examples. While a good *a priori* performance is necessary for a successful application as model, it is not sufficient. Even small *a priori* errors can accumulate over time, finally blowing up a simulation, especially in the context of DNS with less-dissipative solvers. Therefore, a good *a posteriori* performance should always be the target but is also significantly more difficult to achieve. For the generic turbulence and the interfacial jet cases, nondimensionalized quantities are shown, as usually done in literature, while the combustion case presents dimensionalized quantities.

3.1 Decaying Turbulence

The decaying turbulence case was computed using the solver psOpen, which solves the incompressible Navier-Stokes equations formulated in spectral space but with the non-linear term computed in physical space. It uses the library P3DFFT for spatial decomposition and to perform the fast Fourier transformation [43]. The simulations for this test case were computed on JUQUEEN, employing more than 1.8 Million concurrent threads and costing more than 35×10^6 core-h [27]. Turbulence along with scalars are initialized on a uniform grid with 4096^3 cells and an initial turbulence intensity of $u_0'^2 = \frac{2}{3}\langle k \rangle$ with $\langle k \rangle$ as ensemble-averaged turbulent kinetic energy. Over time, the turbulent kinetic energy decays resulting in larger turbulent structures. Four ensembles with statistically similar initial conditions were computed. The maximum Taylor-based Reynoldsnumber is 88.

The decaying turbulence application is a very good baseline case, as it features many different Reynolds numbers over time, which is typical for many practical applications. Therefore, the decay, typically evaluated using the ensemble-averaged turbulent kinetic energy and the ensemble-averaged dissipation rate $\langle \varepsilon \rangle$, must be correctly predicted by the model, even on a much coarser LES mesh. The results presented in this paper for the decaying turbulence case focus on the velocity prediction, and results for scalars are omitted. However, the combustion cases discussed below focus on scalar prediction. A uniform LES mesh of 64^3 was used for the decaying turbulence case [13].

The energy spectrum is a very important measure for quantifying the distribution of turbulent energy among the length scales. The turbulent kinetic energy is produced on the large scales, transported in the inertial range, and dissipates on the small scales. Fig. 3 shows the dimensionless energy spectrum, denoted as $\mathscr{S}^*$, based on all normalized velocity components, for the decaying turbulence case. The spectrum is computed with the fully resolved velocity data, the filtered velocity data, and the reconstructed velocity data, and the wavenumber κ is normalized with the peak wavenumber κ_p of the initial energy spectrum. The filter removes the smallest scales and the corresponding energy contribution. The task of the model is to reconstruct these scales and add energy again. Over the full range of scales, the accuracy of the reconstructed data is very good. The accuracy of the energy spectrum is remarkable for two reasons: First, the PIESRGAN model does not use any explicit information about the shape of the energy spectrum nor performs any kind of forcing in wavenumber space. However, the accuracy in wavenumber space outperforms any other state-of-the-art turbulence model. Second, the energy spectrum is crucial for the behavior of turbulent flows and contains many important information about physical processes, such as the transport of energy from the large to the small scales. The high accuracy in wavenumber space is therefore of utmost importance for evaluating the performance of the PIESRGAN model for turbulence prediction.

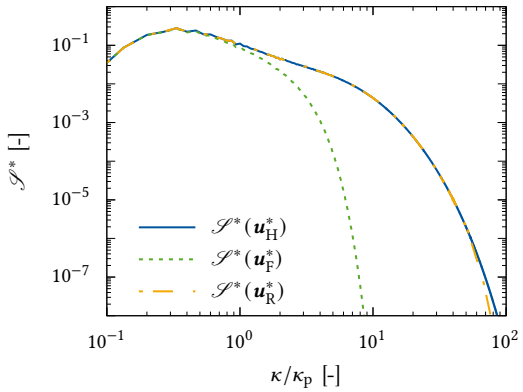

Figure 3: Dimensionless energy spectrum $\mathscr{S}^*$ plotted over normalized wavenumber κ/κ_p and evaluated on fully resolved data, filtered data, and reconstructed data.

Fig. 4 shows the evolution of the ensemble-averaged turbulent kinetic energy and ensemble-averaged dissipation rate over time, evaluated based on the DNS ensemble data and with the PIESRGAN-LES. The decay predicted by the PIESRGAN-LES is in good agreement with the decay of the DNS data. Note, in particular, that the dissipation rate would be significantly off without the sufilter model, as dissipation occurs on the small scales which are not captured by the filtered equations on the coarse mesh. As will be shown in Sec. 3.2, the results are even good, if the turbulence is no longer fully homogeneous. In cases with highly anisotropic turbulence, a combination of homogeneous isotropic turbulence and boundary layer turbulence can be used for training.

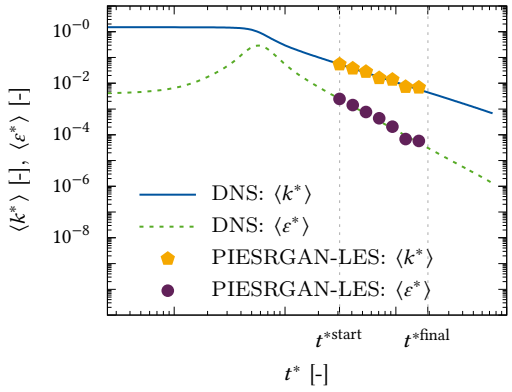

Figure 4: Temporal evolution of the ensemble-averaged dimensionless turbulent kinetic energy $\langle k^* \rangle$ and ensemble-averaged dimensionless dissipation rate $\langle \varepsilon^* \rangle$.

The spatial reduction in simulation size to 64^3, i. e., by multiple orders of magnitude, is remarkable. It emphasizes the universal character of turbulence on the small scales. It is also significant in terms of data storage, as it basically reduces drastically the size of the information which needs to be stored - in addition to that of the trained network, which is small. However, this large reduction should be seen as a theoretical limit, as reductions to meshes about 10 to 20 times coarser per direction seem to be more robust for more complex cases. Furthermore, reconstructing very large parts is also very memory intensive, making a little bit finer meshes with smaller parts to be reconstructed often more efficient. Note that for flow simulations, the time step size is also coupled to the mesh size via the CFL number. Therefore, an increase in spatial cell size allows also more freedom with larger time steps, however, to accurately reproduce the fully resolved data, a time step in-between the DNS time step size and the theoretical new CFL limit was found to be required and a time step size of about 50 % of the theoretical new CFL limit was used here.

3.2 Turbulent Premixed Flame Kernel

The turbulent premixed flame kernel application focuses on the iso-octane/air flame kernel setup under real engine conditions defined in [20–22] with unity Lewis number, i. e., with the same diffusion coefficient for all scalar species. It computes the evolution of an originally spherical flame kernel in a decaying homogeneous isotropic turbulence field on a mesh with 960^3 cells. The cases were computed with the code CIAO on a staggered mesh [8]. The flame kernels were computed in the low-Mach limit using the Curtiss-Hirschfelder approximation [30] for diffusive scalar transport and including the Soret effect. A reaction mechanism with 26 species was used, increasing the computing cost significantly compared to a nonreactive case, as a system of 26 coupled PDEs (cf. (1)) needs to be solved. The cases were originally computed on SuperMUC, however for the present work they were recomputed on the JURECA-DC (Booster) module, as part of the PIESRGAN training process to train on-the-fly. The cost for two realizations of the flame kernel on JURECA-DC (Booster) were about 22×10^6 core-h.

The reaction progress variable ζ is essential for analyzing such premixed combustion cases. Reference [21] defined it as sum of the mass fractions of H_2, H_2O, CO, and CO_2 and introduced a simplified reaction progress variable behaving according to a transport equation (cf. (1)) with the thermal diffusion coefficient as diffusion coefficient and the sum of the source terms of the species used for the definition of the reaction progress variable as chemical source term. The temporal evaluation of one realization of the flame kernel is visualized in the top row of Fig. 5. The initially spherical flame kernel increases significantly in size and deforms. The surface is strongly wrinkled. The training of PIESRGAN was performed with multiple filter stencil widths varying from 5 to 15 cells [11]. Furthermore, Fig. 5 presents reconstruction results for the simplified reaction progress variable, two species mass fractions, and one velocity component. The agreement between fully resolved and reconstructed data is good. The filtered data are less sharp as they feature less small-scale structures due to the filtering over 15 cells.

References [20, 22] pointed out that an accurate prediction of the surface growth is a key for understanding cycle-to-cycle variations (CCVs) with respect to the global heat release in such flame kernel setups. The surface growth is a highly coupled quantity and very difficult to predictively estimate with reduced order models. The a posteriori prediction of the surface density is therefore a perfect metric for evaluating the PIESRGAN-LES prediction quality. The evolution of the flame surface density Σ is shown in Fig. 6 The ensemble-averaged turbulent kinetic energy in the unburnt mixture is given in Fig. 7. To further demonstrate the training and execution processes of PIESRGAN, flame surface densities of three different flame kernels are given, two flame kernels that were used for training and one flame kernel that was used for testing. CCVs among the different runs can be seen, making the accurate prediction of the target flame kernel remarkable. Note that training data are not shown for the turbulent kinetic energy, as all kernel variations used the same initial turbulence field with only different flame kernel start locations. Overall, the agreement between DNS data and PIESRGAN-LES data is good, emphasizing the potential of the introduced PIESRGAN-subfilter modeling approach.

3.3 Interfacial Jet

Reduced order models of interface formation are another important challenge for industrially relevant flows. DNSs of a temporal jet configuration were used to study modeling of interface-driven effects in two-phase flows (cf. Fig. 1) in this work. All jets have a bulk Reynolds number of 5000 and a viscosity ratio of 40. The Weber number, i. e., a measure of the relative importance of the fluid's inertia compared to its surface tension, and the density ratio vary by a factor of 20 between realizations in order to study the sensitivity of the interface with respect to these parameters. As larger Weber numbers require higher spatial resolution, up to about eight billion cells were used for the largest DNS. All DNSs were performed using CIAO along with a 3-D unsplit coupled level set/volume of fluid (3DU-CLSVOF) scheme considering surface tension [25, 38] and a second-order accurate, monotonicity preserving Lagrange-remap solver [39]. Due to the infinitesimally small geometric features during interface formation, corresponding interface simulations are typically mesh dependent and require very fine meshes. This mesh

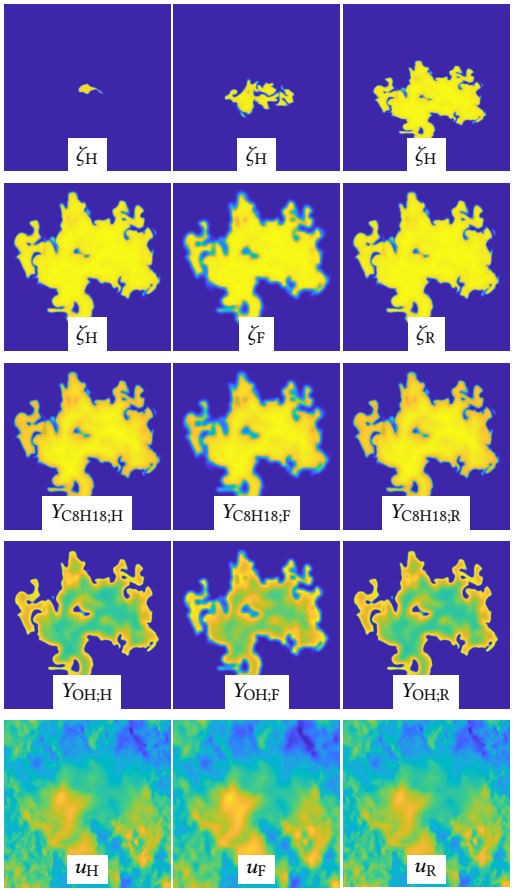

Figure 5: Visualization of one turbulent premixed flame kernel realization. The first row shows the temporal evaluation of the simplified progress variable ζ at 0.06 ms, 0.21 ms, and 0.36 ms. All other figures show a zoomed view of the fully resolved data, filtered data, and reconstructed data for the simplified reaction progress variable ζ, the C_8H_{18} mass fraction, Y_{C8H18}, the OH mass fraction, Y_{OH}, and a velocity component u employing PIESRGAN at 0.36 ms. Colormaps span from blue (minimum) to yellow (maximum).

dependency is significantly reduced by employing the PIESRGAN modeling approach.

PIESRGAN reconstruction was used on the volume of fluid (VoF) field. After reconstruction, surface tension forces are enforced. Thus, the numerical scheme benefits twice from the data-driven reconstruction. First, the interface reconstruction/remapping step results in a more accurate representation of the interface. Second, the size of the surface tension, which depends on the geometrical features of the interface, can be more accurately evaluated. As physically informed contribution to the loss function, the interface jump conditions were enforced along with the continuity term reading

$$L_{\text{physics}} = \beta_{41} L_{\text{mass}} + \beta_{42} L_{\text{jump}}. \tag{6}$$

The a priori test uses data at one time step, filters the data, and reconstructs the data with PIESRGAN. The filtered mesh, which

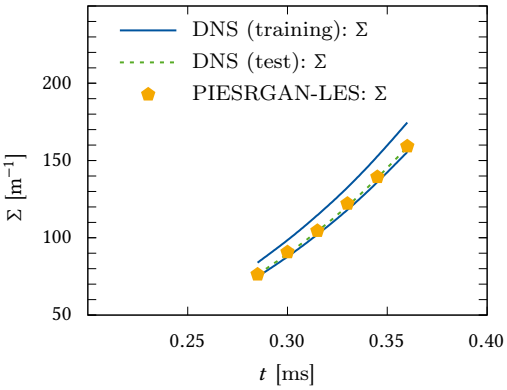

Figure 6: Evolution of the ensemble-averaged turbulent kinetic energy in the unburnt $\langle k_u \rangle$ and the flame surface density Σ over time t.

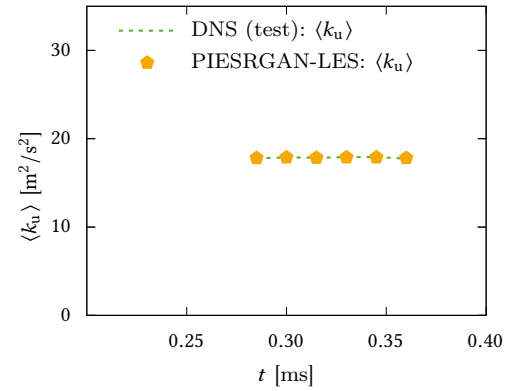

Figure 7: Evolution of the ensemble-averaged turbulent kinetic energy in the unburnt $\langle k_u \rangle$ and the flame surface density Σ over time t.

was chosen, used half of the cells per direction compared to the DNS mesh. The results are shown in Fig. 8. The visual agreement is very good.

The droplet, which is formed and already visible in the a priori visualization, is tracked over time with DNS and LES with a coarser mesh for an a posteriori test. The result is shown in Fig. 9. Again, the visual agreement is very good. Without model, the breakup on the coarser mesh would be slower.

4 CONCLUSIONS

The presented results show that PIESRGAN is able to accurately model various configurations run on current supercomputers and to significantly accelerate complex simulation workflows. The simple decaying turbulence case was run with only 201×10^3 core-h, and also the more complex simulations were accelerated significantly. The speedup could be used to create large ensembles very efficiently, e. g., to quantify CCVs. Projecting the current development of GPU performance, it is expected that the cost for PIESRGAN will further

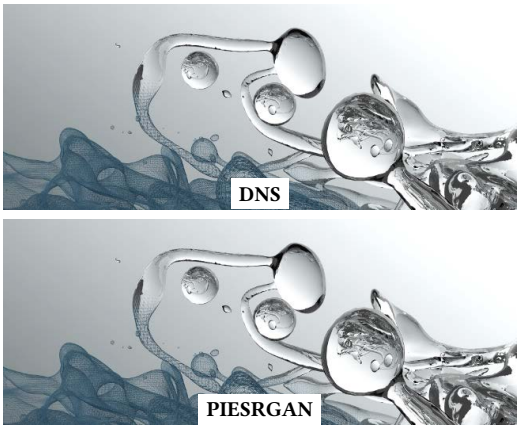

Figure 8: A priori results for the VoF field shown for DNS and reconstructed data.

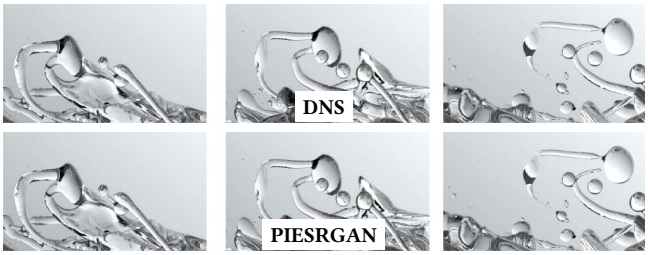

Figure 9: A prosteriori results for the VoF field shown for DNS and LES.

decrease, making this approach even more promising. As demonstrated, the same architecture can be used to systematically develop reduced order models for multi-physics flows by only choosing an appropriate physics-informed loss function term.

The presented coreh-to-solution comparisons between DNS and PIESRGAN-LES are obviously only one important metric. Comparisons between PIESRGAN-LES and classical LES, such as the dynamic Smagorinsky model [47], are also interesting but more difficult to pursue. The PIESRGAN-LES presented in this paper is highly optimized, and thus comparisons with classical LES without this degree of case-specific optimization may be biased towards the PIESRGAN-LES results. On the other hand, using the same meshes for PIESRGAN-LES and classical LES is biased towards classical LES, as PIESRGAN-LES is more robust to underresolved flow situations [13], but accuracy measurements are also often non-trivial. Finally, there are many cases where classical LES is not able to achieve acceptable results at all. In general, studies have shown that PIESRGAN LES are significantly more expensive per time step than classical LES on similar meshes. Since they require less resolution to achieve the same accuracy, mesh resolution can be used to reduce this cost overhead. Moreover, PIESRGAN-LES can reproduce DNS quantities with high accuracy at significantly lower cost than DNS, which are not predictable at all with classical LES models. This is a strong advantage depending on the target case.

Besides the accuracy and speedup, the PIESRGAN-subfilter approach features additional advantages in the context of recent supercomputers. First, most world-leading supercomputers gain most of their performance from their GPUs. The focus on GPUs compared to CPUs allowed to setup the world's first Exascale machines but this development is also very critical for applications which are difficult to port to GPUs. Flow simulations with their typical stencil sizes are among those critical simulations, which significantly limits their performance on the newest generation of supercomputer setups. PIESRGAN could be a workaround as it smoothly runs on GPUs and basically shifts CPU-cost to GPUs by reducing the cells computed on the LES mesh and adding extra cost during the reconstruction. Second, current supercomputer workflows often require data-centric approaches, as moving data is more expensive than computing data. PIESRGAN reduces the amount of moved data for the cost of additional computing operations, which can be advantageous. It can be also used as coupler in the context of modular supercomputing, reducing the amount of data shifted between multiple clusters significantly without lacking accuracy. Finally, PIESRGAN-LES significantly reduces the amount of I/O, due to much smaller size of the filtered mesh. As I/O often accounts for about 10 % of computing cost in simulations, this is a non-negligible advantage.

The turbulence, turbulent combustion, and interface application cases presented in this work were carefully chosen examples to showcase the modeling performance of AI-based super-resolution in fluid dynamics and emphasize its HPC potential.

ACKNOWLEDGMENTS

The author acknowledges computing time grants for the projects JHPC55 and TurbulenceSL by the JARA-HPC Vergabegremium provided on the JARA-HPC Partition part of the supercomputer JURECA at Jülich Supercomputing Centre, Forschungszentrum Jülich, the Gauss Centre for Supercomputing e.V. (www.gauss-centre.eu) for funding this project by providing computing time on the GCS Supercomputer JUWELS at Jülich Supercomputing Centre (JSC), and funding from the European Union's Horizon 2020 research and innovation program under the Center of Excellence in Combustion (CoEC) project, grant agreement no. 952181.

REFERENCES

[1] M. Bode. 2022. Applying physics-informed enhanced super-resolution generative adversarial networks to finite-rate-chemistry flows and predicting lean premixed gas turbine combustors. *arXiv preprint arXiv:2210.16219* (2022).
[2] M. Bode. 2022. Applying physics-informed enhanced super-resolution generative adversarial networks to large-eddy simulations of ECN Spray C. *SAE International Journal of Advances and Current Practices in Mobility* 4 (2022), 2211–2219. Issue 6.
[3] M. Bode. 2022. Applying physics-informed enhanced super-resolution generative adversarial networks to turbulent non-premixed combustion on non-uniform meshes and demonstration of an accelerated simulation workflow. *arXiv preprint arXiv:2210.16248* (2022).
[4] M. Bode. 2023. AI super-resolution: Application to turbulence and combustion. In *Machine learning and its application to reacting flows, Lecture Notes in Energy 44*, N. Swaminathan and A. Parente (Eds.). Springer.
[5] M. Bode. 2023. AI super-resolution-based subfilter modeling for finite-rate-chemistry flows: A jet flow case study. *SAE Technical Paper 2023-01-0200* (2023).
[6] M. Bode, N. Collier, F. Bisetti, and H. Pitsch. 2019. Adaptive chemistry lookup tables for combustion simulations using optimal B-spline interpolants. *Combustion Theory and Modelling* 23, 4 (2019), 674–699.
[7] M. Bode, M. Davidovic, and H. Pitsch. 2019. Towards clean propulsion with synthetic fuels: Computational aspects and analysis. In *High-Performance Scientific*

Computing. Springer Nature, 185–207.

[8] M. Bode, A.Y. Deshmukh, T. Falkenstein, S. Kang, and H. Pitsch. 2023. Hybrid scheme for complex flows on staggered grids and application to multiphase flows. *J. Comput. Phys.* 474 (2023), 108478.

[9] M. Bode, F. Diewald, D. Broll, J. Heyse, et al. 2014. Influence of the injector geometry on primary breakup in diesel injector systems. *SAE Technical Paper 2014-01-1427* (2014).

[10] M. Bode, T. Falkenstein, M. Davidovic, et al. 2017. Effects of cavitation and hydraulic flip in 3-hole GDI injectors. *SAE International Journal of Fuels and Lubricants* 10, 2 (2017), 380–393.

[11] M. Bode, M. Gauding, D. Goeb, T. Falkenstein, and H. Pitsch. 2023. Applying physics-informed enhanced super-resolution generative adversarial networks to turbulent premixed combustion and engine-like flame kernel direct numerical simulation data. *Proceedings of the Combustion Institute* (2023).

[12] M. Bode, M. Gauding, K. Kleinheinz, and H. Pitsch. 2019. Deep learning at scale for subgrid modeling in turbulent flows: regression and reconstruction. *Lecture Notes in Computer Science* 11887 (2019), 541–560.

[13] M. Bode, M. Gauding, Z. Lian, D. Denker, M. Davidovic, K. Kleinheinz, et al. 2021. Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows. *Proceedings of the Combustion Institute* 38 (2021), 2617–2625.

[14] W. T. Chung, A. A. Mishra, N. Perakis, and M. Ihme. 2021. Data-assisted combustion simulations with dynamic submodel assignment using random forests. *Combustion and Flame* 227 (2021), 172–185.

[15] D. Denker, A. Attili, M. Gauding, K. Niemietz, M. Bode, and H. Pitsch. 2021. A new modeling approach for mixture fraction statistics based on dissipation elements. *Proceedings of the Combustion Institute* 38 (2021), 2681–2689.

[16] O. Desjardins, G. Blanquart, G. Balarac, and H. Pitsch. 2008. High order conservative finite difference scheme for variable density low Mach number turbulent flows. *J. Comput. Phys.* 227, 15 (2008), 7125–7159.

[17] C. Dong, C. C. Loy, K. He, and X. Tang. 2014. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*. Springer, 184–199.

[18] C. Dong, C. C. Loy, K. He, and X. Tang. 2015. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 2 (2015), 295–307.

[19] Giuseppe D'Alessio, Alessandro Parente, Alessandro Stagni, and Alberto Cuoci. 2020. Adaptive chemistry via pre-partitioning of composition space and mechanism reduction. *Combustion and Flame* 211 (2020), 68–82.

[20] T. Falkenstein, H. Chu, M. Bode, S. Kang, and H. Pitsch. 2020. The role of differential diffusion during early flame kernel development under engine conditions - Part II: Effect of flame structure and geometry. *Combustion and Flame* 221 (2020), 516–529.

[21] T. Falkenstein, S. Kang, L. Cai, M. Bode, and H. Pitsch. 2020. DNS study of the global heat release rate during early flame kernel development under engine conditions. *Combustion and Flame* 213 (2020), 455–466.

[22] T. Falkenstein, A. Rezchikova, R. Langer, M. Bode, S. Kang, and H. Pitsch. 2020. The role of differential diffusion during early flame kernel development under engine conditions - Part I: Analysis of the heat-release-rate response. *Combustion and Flame* 221 (2020), 502–515.

[23] S. Farazi, J. Hinrichs, M. Davidovic, T. Falkenstein, et al. 2019. Numerical investigation of coal particle stream ignition in oxy-atomsphere. *Fuel* 241 (2019), 477–487.

[24] U. Frisch. 1995. *Turbulence - The legacy of A. N. Kolmogorov*. Cambridge University Press, Cambridge, UK.

[25] F. Fröde, T. Grenga, V. Le Chenadec, M. Bode, and H. Pitsch. 2022. A three-dimensional cell-based volume-of-fluid method for conservative simulations of primary atomization. *J. Comput. Phys.* 465 (2022), 111374.

[26] K. Fukami, R. Maulik, N. Ramachandra, K. Fukagata, and K. Taira. 2021. Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning. *Nature Machine Intelligence* 3 (2021), 945–951.

[27] M. Gauding, L. Wang, J. H. Göbbert, M. Bode, L. Danaila, and E. Varea. 2019. On the self-similarity of line segments in decaying homogeneous isotropic turbulence. *Computers & Fluids* 180 (2019), 206–217.

[28] I.J. Goodfellow, J. Pouget-Agadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. Generative Adversarial Networks. *arXiv preprint arXiv:1406.2661* (2014).

[29] M. T. Henry de Frahan, S. Yellapantula, R. King, M. S. Day, and R. W. Grout. 2019. Deep learning for presumed probability density function models. *Combustion and Flame* 208 (2019), 436–450.

[30] J.O. Hirschfelder, C.F. Curtiss, R.B. Bird, and M.G. Mayer. 1964. *Molecular theory of gases and liquids*.

[31] J. Johnson, A. Alahi, and L. Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*. Springer, 694–711.

[32] A. Jolicoeur-Martineau. 2018. The relativistic discriminator: a key element missing from standard GAN. *arXiv preprint arXiv:1807.00734* (2018).

[33] J. Kim, J. Kwon Lee, and K. Mu Lee. 2016. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1646–1654.

[34] J. Kim, J. Kwon Lee, and K. Mu Lee. 2016. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1637–1645.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.

[36] W.-S. Lai, J.-B. Huang, N. Ahuja, and booktitle=Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition pages=624–632 year=2017 Yang, M.-H. [n.d.]. Deep Laplacian pyramid networks for fast and accurate super-resolution.

[37] C. J. Lapeyre, A. Misdariis, N. Cazard, D. Veynante, and T. Poinsot. 2019. Training convolutional neural networks to estimate turbulent sub-grid scale reaction rates. *Combustion and Flame* 203 (2019), 255–264.

[38] V. Le Chenadec and H. Pitsch. 2013. A 3d unsplit forward/backward volume-of-fluid approach and coupling to the level set method. *J. Comput. Phys.* 233 (2013), 10–33.

[39] V. Le Chenadec and H. Pitsch. 2013. A monotonicity preserving conservative sharp interface flow solver for high density ratio two-phase flows. *J. Comput. Phys.* 249 (2013), 185–203.

[40] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4681–4690.

[41] Y. Li, Y. Ni, R. A. C. Croft, T. Di Matteo, S. Bird, and Y. Feng. 2021. AI-assisted superresolution cosmological simulations. *Proceedings of the National Academy of Sciences* 118 (2021), e2022038118. Issue 19.

[42] A. L. Maas, A. Y. Hannun, and A. Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. *Proceedings of the 30th International Conference on Machine Learning* 30 (2013).

[43] D. Pekurovsky. 2012. P3DFFT: A framework for parallel computations of Fourier transforms in three dimensions. *SIAM Journal on Scientific Computing* 34 (2012), 192–209. Issue 4.

[44] H. Pitsch. 2006. Large-eddy simulation of turbulent combustion. *Annual Review of Fluid Mechanics* 38 (2006), 453–482.

[45] S. B. Pope. 2000. *Turbulent flows*. Cambridge University Press, Cambridge, UK.

[46] K. Simonyan and A. Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[47] J. Smagorinsky. 1963. General circulation experiments with the primitive equations: I. The basic experiment. *Monthly Weather Review* 91, 3 (1963), 99–164.

[48] K. Stengel, A. Glaws, D. Hettinger, and R. N. King. 2020. Adversarial super-resolution of climatological wind and solar data. *Proceedings of the National Academy of Sciences* 117 (2020), 16805–16815. Issue 29.

[49] Y. Tai, J. Yang, X. Liu, and C. Xu. 2017. MemNet: A persistent memory network for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision*. 4539–4547.

[50] G. Tryggvason, R. Scardovelli, and S. Zaleski. 2011. *Direct numerical simulations of gas-liquid multiphase flows*. Cambridge University Press, Cambridge, UK.

[51] K. Wan, S. Hartl, L. Vervisch, P. Domingo, R. S. Barlow, and C. Hasse. 2020. Combustion regime identification from machine learning trained by Raman/Rayleigh line measurements. *Combustion and Flame* 219 (2020), 268–274.

[52] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy. 2018. ESRGAN: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision*.

[53] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. 2018. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision*. 286–301.

Received XXX; revised XXX; accepted XXX