A Multi-fidelity Double-Delta Wing Dataset and Empirical Scaling Laws for GNN-based Aerodynamic Field Surrogates

Yiren Shen* and Juan J. Alonso†
Stanford University, Stanford, CA 94305, USA

Data-driven surrogate models are increasingly adopted to accelerate vehicle design. However, open-source multi-fidelity datasets and empirical guidelines linking dataset size to model performance remain limited. This study investigates the relationship between training data size and prediction accuracy for a graph neural network (GNN) based surrogate model for aerodynamic field prediction. We release an open-source, multi-fidelity aerodynamic dataset for double-delta wings, comprising 2448 flow snapshots across 272 geometries evaluated at angles of attack from 11° to 19° at $Ma = 0.3$ using both vortex lattice method (VLM) and Reynolds-Averaged Navier-Stokes (RANS) solvers. The geometries are generated using a nested Saltelli sampling scheme to support future dataset expansion and variance-based sensitivity analysis. Using this dataset, we conduct a preliminary empirical scaling study of the `MF-VortexNet` surrogate by constructing six training datasets with sizes ranging from 40 to 1280 snapshots and training models with 0.1 to 2.4 million parameters under a fixed training budget. We find that the test error decreases with data size with a power-law exponent of -0.6122, indicating efficient data utilization. Based on this scaling law, we estimate that the optimal sampling density is approximately eight samples per dimension in a d-dimensional design space. The results also suggest improved data utilization efficiency for larger surrogate models, implying a potential trade-off between dataset generation cost and model training budget.

I. Introduction

Surrogate models are widely used in aircraft design and multidisciplinary design optimization (MDO) due to the complex nature of underlying physics and engineering problems [1]. As a result of the complexity, the assessments and optimizations of vehicles are costly when high-dimensional models or experiments are used as the means of performance assessment. The core idea of surrogate modeling is to construct a mathematically simplified function that captures a predefined set of quantity of interests (QOI), such as lift and drag coefficients C_L and C_D, allowing the surrogate to replace costly high order evaluations [2]. By leveraging data-driven approximations, surrogate models enable rapid predictions while preserving essential physical trends, thereby significantly reducing the cost of design and analysis and accelerating the new-product introduction (NPI) process.

Recently, due to the advances in computational hardware and novel data-driven models, such as neural networks, a wide variety of new surrogate model methods has been proposed for aerospace applications [3–11]. These surrogate modeling techniques range from Gaussian Process Regression (GPR), Polynomial Chaos Expansion (PCE), Proper Orthogonal Decomposition (POD) and reduced order modeling (ROM), Deep Neural Networks (DNNs), to Physics Informed Neural Networks (PINNs). Each class of method is suited to different trade-offs among prediction accuracy, applicability, interpretability, and computational efficiency. Some methods, such as GPR and PCE, are more suitable for the prediction of a set of scalar quantities, while others, such as POD, PINNs, and DNNs, have the potential for predicting distributed field quantities such as pressure or velocity fields.

The capability of offering complex mappings from design variables and flow conditions to spatially resolved fields is often desired in MDO cycles. These field quantities provide richer design feedback for designers, allowing the coupling of analyses across disciplines and fidelities with multiple solvers, and enabling postprocessing of arbitrary QOIs. Popular techniques for field surrogate modeling include POD-Kriging [12] and convolutional neural network (CNN) [13]. These methods often require a consistent spatial topology among surrogate data inputs; however, when the geometry undergoes large shape changes, maintaining such correspondence becomes difficult, thereby limiting their generalizability across the design space in MDO applications.

*Ph.D. Candidate, Department of Aeronautics and Astronautics, AIAA Student Member.
†Vance D. and Arlene C. Coffman Professor, Department of Aeronautics and Astronautics, AIAA Fellow.

To overcome this limitation, recent work has shifted toward unstructured geometry representations that operate directly on unstructured point clouds and graphs. Point-cloud-based models naturally encode both geometric and solution information as nodal or edge features, eliminating the need for mesh mapping. For example, `PointNet` architecture directly predicts surface pressure distributions from point clouds [14], while graph neural network (GNN) surrogates such as `MeshGraphNets` demonstrate accurate field reconstruction across widely varying geometries [15]. In parallel, Implicit Neural Representations have been used for flow-field reconstruction [16], while operator-learning frameworks such as DeepONet and DoMINO [17, 18], Geometry-Informed Neural Operators [19], and Fourier Neural Operators (FNO) [20] have been applied to aerodynamics prediction and parametric flow modeling [19–21], providing mesh-independent mappings between function spaces. More recently, transformer-based surrogates have also been proposed for vehicle aerodynamics modeling [22]. Collectively, these developments signal a broader transition from mesh-dependent field surrogates toward geometry- and discretization-agnostic surrogate models for aerodynamic design.

Beyond geometric generalization, it is also desirable that surrogate models support mappings across fidelity hierarchies. In NPI process, it is common for analyses to be performed at different fidelities: from the initial conceptual design to detailed design, increasingly sophisticated higher-fidelity models are deployed for assessment and optimization. It is therefore desirable to have a mean of bridging the fidelity gaps across tools. Among the field-prediction approaches, graph-based physics-informed surrogates have shown particular promise for multi-fidelity mappings as these models are relatively inexpensive to train and the graph-based contextualization of data sources fits mesh-based analyses naturally. Previously, we proposed `MF-VortexNet`, a physics-informed multi-fidelity GNN surrogate that learns to map low-fidelity pressure fields computed by VLM to high-fidelity pressure fields computed by computational fluid dynamics (CFD) on the same lifting-surface lattice [10]. When applied to low-speed, high angle-of-attack (AOA) Delta wing configurations featuring nonlinear fluid dynamics that are not modeled by the VLM solver, `MF-VortexNet` successfully maps low-fidelity pressure loading to its high-fidelity counterpart and thus significantly reduces normalized errors in aerodynamic coefficients prediction, while generalizing reasonably across unseen geometries within a prescribed design space [23]. When integrated with a conceptual design environment, the model showed sufficient fidelity gain to enable stability-constrained aerodynamic shape optimization for a double-delta (cranked delta) wing test case [24].

However, questions remain regarding how `MF-VortexNet`'s prediction accuracy scales with training dataset size and what data requirements are necessary to reach a target accuracy level. In particular, empirical scaling laws [25] for physics-informed, multi-fidelity graph neural network (GNN) surrogates models for field prediction remain largely absent from the aerodynamic literature [26], hindering informed decisions about design of experiments (DOE) strategies and the application of similar tools in MDO.

To study how surrogate-model accuracy scales with dataset size, the underlying training dataset must contain both substantial geometric variability and sufficiently complex flow physics. Although the AIAA Applied Aerodynamics Surrogate Modeling (AASM) discussion group has recently released several benchmark datasets for surrogate modeling[27], these datasets do not provide the controlled geometric parametrization desired for the present scaling analysis. To address this need, we adopt a low-speed double-delta wing as the test configuration. This geometry offers a six-dimensional design space that enables systematic shape variations. Furthermore, the external flow over a Delta wing at higher AOAs is highly nonlinear and dominated by vortex-driven phenomena [28], making it an informative and challenging setting for evaluating the predictive capacity of field-based surrogate models. Finally, the chosen geometry maintains a balance between engineering realism and data-preprocessing simplicity: the outer mold line (OML) surfaces require no complex cleanup or fuselage-wing intersection handling, yet still represent a three-dimensional aerodynamic problem.

This work addresses these two interrelated gaps. First, we present an open-source, multi-fidelity aerodynamic dataset for double-delta wings across 272 unique geometries and multiple AOAs ranging from 11° to 19° at low speed ($Ma = 0.3$). The DOE employs a nested Saltelli sampling scheme to generate geometric configurations, enabling nested geometric variations and supporting future variance-based sensitivity analysis and uncertainty quantification. Aerodynamic coefficients, surface solutions, and volume solutions are provided in open formats, along with the CFD configuration files, to maximize downstream usability of the dataset. Second, we perform an empirical scaling study of the `MF-VortexNet` model using nested subsets of the dataset. By selecting between 8 and 256 geometric configurations for training set construction, we quantify a power-law decay of prediction error with increasing sampling density in the design space.

The remainder of this paper is organized as follows. Section II outlines the relevant technical framework of the different analysis components, including geometry generation environment, low- and high-fidelity solvers, and the `MF-VortexNet` surrogate model. Section III presents a detailed description of the double-delta wing aerodynamic dataset, including the design space, DOE, simulation configurations, and simulation results. Section IV presents the

setup and results of the scaling study. Finally, Section V discusses the implications of the current study and outlines future work.

II. Methodology

As this study involves modeling external flow over wings with multiple solvers at different fidelities, and utilizing a machine learning model `MF-VortexNet` to learn the mapping from a pressure–geometry–flow-condition state to a high-fidelity pressure field, this section provides a brief discussion of the modules used throughout the study.

A. Multi-fidelity Geometry Representation

The `SUAVE` conceptual design environment [29] is used to generate vehicles from geometry parameters, drive low-fidelity simulations, and create surface-mesh tessellations using its built-in `OpenVSP` [30] interface. Depending on the subsequent applications, the model representations of a given vehicle configuration may have different fidelities.

At the lowest fidelity, an instance of the `Vehicle` class in `SUAVE` is created, taking in the corresponding design variable vector $\mathbf{x} \in \mathbf{R}^n$, where n is the dimension of the design space, and a prescribed process that constructs a vehicle from $\mathbf{x}$. Details of the computer program structure, including the `Vehicle` class definition, can be found in Lukaczyk et al. [29]. When $\mathbf{x}$ is changed, different instances are created to represent different vehicle configurations.

When different analyses are conducted on the vehicle instance, different types of instance representation are needed. When the VLM (see Section II.B) is utilized, a lattice grid of the vortex distribution is created based on the `Vehicle` instance by incorporating the encoded geometric parameters. Additional configuration settings, such as the number of panels or the distribution functions used for the lattice discretization, may change the vehicle-geometry representation for a given instance. The lattice grid and vortex distribution are used by the VLM solver to perform aerodynamic analyses.

Likewise, for CFD simulations, a representation of the OML of the vehicle is required by the solver. Under such analyses, we use `SUAVE`'s `OpenVSP` [30] interface to translate the `SUAVE` instance to an OML boundary representation. Subsequently, `OpenVSP`'s `ComputeCFDMesh` function is used to generate a tessellated surface mesh. This mesh is then imported into `Pointwise`, together with a far-field boundary defined with a diameter of 40 mean aerodynamic chords (MACs), to define the fluid domain. The fluid domain is then further discretized to generate the volume mesh for CFD analyses.

B. Low-fidelity Solver

The VLM solver solves the potential-flow problem for the perturbation velocity potential, subject to a no-penetration boundary condition on the lifting surface. The lifting surface is discretized into a quadrilateral panel lattice. Horseshoe vortices are defined on each panel's quarter-chord as well as its side bounds, with semi-infinite trailing legs extending downstream. The no-penetration boundary condition is enforced at the collocation points located at the center of three-quarter-chord of each panel. This leads to a linear system for the vortex strength of each panel's horseshoe vortex, which can be solved. The resultant vortex strengths at each panel can then be summed for induced-velocity computation using the Biot-Savart law [31]. The pressure perturbation due to the induced velocity is computed using Bernoulli's theorem [31]. The specific VLM used in this work is shipped with `SUAVE`, which is based on Miranda et al.'s `VORLAX` implementation, and is suitable for compressibility-corrected, inviscid, attached flows [32].

Due to the compressibility correction, the VLM predictions for vehicles at high speed and low AOA is relatively accurate. The computational cost, at a chordwise-by-spanwise lattice of 30×32, is negligible. In addition, the solver's convergence is robust across a wide design space, making it ideal for rapid design space exploration studies. However, due to the neglect of nonlinear flow effects such as vortex lift and separation, the prediction accuracy deteriorates for vehicles at low speed under high AOAs. As shown by Shen and Alonso in their previous study comparing the QOIs computed from VLM and RANS CFD, VLM results for relevant Delta wing configurations are relatively accurate up to AOA of 10° [23].

C. High-fidelity Solver

The `SU2` RANS CFD solver is used to compute aerodynamics [33]. The solver solves the RANS equations, in which nonlinear flow effects are accounted for. For turbulence modeling, the Spalart-Allmaras turbulence model with rotation corrections (SA-R) [34] is used. The SA-R turbulence model adds a one-equation eddy-viscosity closure to the RANS equations, such that the eddy viscosity is adjusted across different flow regions based on the dominance of rotational and strain effects.

For the present application, the selected solver and turbulence model achieve a balance between the desired accuracy and cost. The chosen solver captures viscous effects and vortex systems for flow that is attached or mildly separated, which constitutes the majority of the flow conditions considered in the current study. Several previous works have demonstrated the suitability of using the SA-R RANS solver for external-flow modeling with strong vortex presence [34, 35]. For Delta wings and double-delta wings, several studies have highlighted the capability of SA model family to predict vortex strength and breakdown [36], to model aerodynamics at higher AOA with accurate capture of vortex-vortex interactions [35], as well as to capture vortex lift-induced effects on surface pressure distribution over wings [28, 36–38].

The chosen SA-R RANS formulation performs well for attached and mildly separated flows, but its accuracy degrades in regimes dominated by massive separation or unsteady fluid dynamics. For double-delta wings, these effects become significant near the critical AOAs at approximately 18° for some geometries, such that vortex sheet oscillation or separated flow are seen. As a result, higher-fidelity approaches such as detached-eddy simulation would be required to achieve reliable predictions. However, the computational cost of these methods is prohibitive to the authors for the broad design-space sampling required in the present study. Consequently, they are not included in the current dataset but represent a potential avenue for future multi-fidelity dataset construction.

D. Multi-fidelity Surrogate Model: `MF-VortexNet`

A surrogate model is deployed to map low-fidelity panel-wise pressure coefficients solved by the VLM solver to the corresponding pressure field of the CFD solver on the same lifting-surface lattice. The surrogate model is `MF-VortexNet` [10], a light-weight physics-informed GNN machine-learning model. To contextualize field data, the VLM lattice is represented by a graph, where nodes aggregate local geometric descriptors, flow conditions, and low-fidelity pressure-coefficient features, while edges encode spatial distances among panels. Within this framework, `MF-VortexNet` employs a U-Net-like encoder-decoder architecture built from multi-head graph-attention convolutional blocks [39] to construct latent representations that capture pressure loading induced by nonlinear vortical structures and other complex flow phenomena, and then decodes these representations to corrected nodal pressure fields that approximate CFD solutions while maintaining robust prediction accuracy across a design space that is typical in the conceptual-design phase.

The surrogate model is embedded into a multi-fidelity aerodynamic workflow as described by Shen et al. to construct paired low- and high-fidelity fields for offline training [10]. In online assessment, the baseline fields computed by the VLM solver are subsequently corrected by `MF-VortexNet`, thereby yielding "quasi-CFD" pressure distributions. Subsequently, derived force and moment coefficients can be computed from the augmented pressure loading at a computational cost comparable to that of the underlying VLM solver. Because the model operates on full fields rather than on a small set of scalar QOIs, arbitrary integrated metrics can be obtained by post-processing the corrected pressure distribution, which improves both flexibility and interpretability of early-stage design studies, and provides a means to couple with solvers in other disciplines. When applied to low-speed, high AOA double-delta wing configurations, the `MF-VortexNet`-augmented workflow demonstrates substantial design-fidelity improvement in capturing moment and stability performance relative to relying on VLM alone, while maintaining generalization across unseen geometries and freestream conditions within a prescribed design space [23, 24].

III. Dataset

This section details the double-delta wing aerodynamic dataset. We first define the parametric double-delta wing geometry and its six-dimensional design space. Next, we describe the DOE, which utilizes a nested Saltelli sampling scheme to ensure scalable sampling. Finally, we present our numerical simulation framework and results, including the automated meshing pipeline and solver configurations for generating the paired VLM and CFD data.

A. Geometry Definition

Six design variables are used to define the double-delta wing shape. Figure 1 shows the half-wing about the central-body symmetry plane. The geometry has six design parameters, including B, the wing span (in inches); BW2, the fractional location of the inboard-to-outboard break-chord relative to the semi-span; SW1 (Λ_{in}), the inboard leading-edge sweep angle; SW2 (Λ_{out}), the outboard leading-edge sweep angle; SR2 (Λ_{te}), the outboard trailing-edge sweep angle; and DROOP (δ), the camber-deflection angle applied to the baseline airfoil. The baseline airfoil is constructed from the root-airfoil geometry of the RW23 wing presented by Wiersema et al. [40] and is applied uniformly across the span.

To allow a richer representation of camber variation within the design space, δ acts by deflecting the baseline airfoil's

leading edge camber about the 30% chord from leading edge. The airfoil morphing process begins with cubic-spline fitting of the baseline airfoil's upper and lower surfaces from their discrete coordinates. The camber and thickness distributions are then extracted on a shared chordwise grid. The droop morphing is subsequently applied to the baseline camber, introducing C^1 and C^2 parametric discontinuities [41] at the pivot point and producing two separate Bézier curves for the leading and trailing cambers, respectively. To ensure smoothness, the Bézier control-point parameters for the leading camber line are solved using a sequential least-squares programming (SLSQP) solver in `SciPy` to minimize the C^1 and C^2 continuity residuals between the leading- and trailing-camber curves at the pivot point. The modified camber is then combined with the original thickness distribution to reconstruct the upper and lower surfaces, with an additional round of smoothing applied to both surfaces' curves.

The spanwise twist distribution is defined as follows: the inboard section, from the root to the break-chord location, maintains a constant twist of 1.5° leading edge up, while the outboard section applies a linear washout from 1.5° to −1.5° toward the tip.

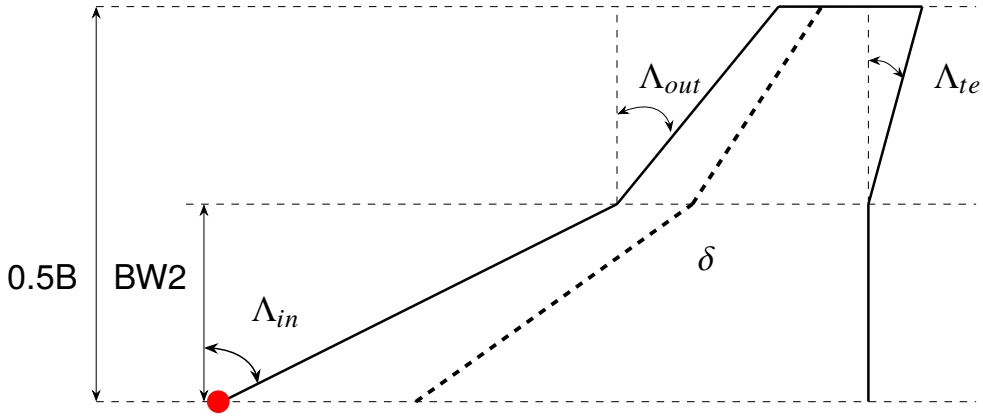


Fig. 1 Geometry definition and its design variables.

B. Design Space

The design space $\mathbf{x}$ is defined as $\mathbf{x} = [\delta, \Lambda_{in}, \Lambda_{out}, \Lambda_{te}, BW2, B]^T, x_i \in [x_i^{min}, x_i^{max}]$, where x_i^{min} and x_i^{max} are the bounds. The selection of the design variable bounds is based on first referencing the design space of recent commercial supersonics technology program, including Supersonic Configurations at Low Speeds (SCALOS) [42], and High-Speed Civil Transport (HCST) [43]. From these studies, the nominal inboard and outboard leading edge sweep (Λ_{in} and Λ_{out}) is 68° and 45° respectively. The aspect ratio of the wing platform is taken to be 2.5. From these nominal values, we consider the design space included in SCALOS program [42], and adjust the design variable bounds so that the created design space does not produce non-physical designs. Specifically, design variable combinations that lead to negative tip chord under our DOE. This constraint reduces the design space represented in this dataset, resulting in a Λ_{in} upper bound smaller than those explored in the SCALOS program. However, the constraint ensures that the resulting samples are balanced, as no non-physical designs are removed from the samples, in contrast to our previous experiments [24]. This design space is presented in Table 1.

C. Design of Experiments

1. Sampling Methods

The DOE involves the selection of combinations of design variables to generate geometry samples. We have three major goals for the DOE. Firstly, we aim to achieve efficient and homogeneous design-space filling, such that the sampling scheme does not introduce sampling bias that leads to clustered design samples within the design space, which could bias the performance of subsequently trained surrogate models. Secondly, the sampling scheme should produce an extendable dataset, such that when the dataset needs to be expanded, new sampling points can be added without requiring existing experiments to be re-run. This property is particularly important when the experiment is costly, such

Design Variable	Lower	Upper	Nominal
δ (°)	-8	7	0
Λ_{in} (°)	47	67	65
Λ_{out} (°)	35	60	45
Λ_{te} (°)	0	25	10
BW2	0.32	0.47	0.4
B (inches)	1100	1300	1200

Table 1 List of design variables and their corresponding upper and lower bounds.

as the CFD simulations in this study. Finally, we want the sampling scheme to support future downstream assessments of the surrogate models trained from the dataset, including applications involving sensitivity analyses or uncertainty quantification.

Popular sampling methods are considered, including factorial sampling, random sampling, Latin Hypercube Sampling (LHS), and quasi-random sequences. The quasi-random sequence sampling methods, particularly Sobol sequence, meet the previously-mentioned DOE goals better. Sobol sequence constructs a sequence of spatial quadrature points that enable fast convergence of n-dimensional integral, that is, to find a sequence x_i with length P such that,

$$\frac{1}{P}\sum_{i=0}^{P-1} f(x_i), \tag{1}$$

approximates the integration

$$\int_{[0,1]^n} f(x)dx. \tag{2}$$

Solving for the Sobol sequence directly is costly, and modern software packages use fast Sobol sequence algorithms in which subsequent Sobol numbers are generated by XOR-ing previous Sobol number with a predefined table of direction numbers [44]. For the current DOE, we adopt `SALib` [45], which contains a Sobol sequence generation routine by Joe and Kuo [44]. For a required P samples in n dimensional space, the Sobol sequence generator generates a base matrix S of size $(P + \text{skipvalue}) \times 2n$. The skipvalue is applied to discard the first several rows of the Sobol sequence matrix, as repetitions may occur [46], thereby improving the uniformity of samples generated. For a predefined skipvalue and problem dimension, the generated Sobol sequence is deterministic. Throughout the experiments, we set the skipvalue to 16.

From the base matrix, we follow Saltelli's sampling scheme [47] using `SALib`'s `sample.saltelli` function with function call of `saltelli.sample(problem, skipvalue=16, calc_second_order=False)`. The process firstly constructs two sampling matrices A and B, such that

$$A[i,:] \leftarrow S[i, 0:n] \quad \text{and} \quad B[i,:] \leftarrow S[i, n:2n]. \tag{3}$$

Then assembling n cross-sampling matrices $AB^{(k)}$ by cross swapping columns of B to a copy of the A matrix. Mathematically, this is equivalent to,

$$AB^{(k)} = (A[:,1], ..., A[:,k-1], B[:,k], A[:,k+1], ..., A[:,n]) \text{ for } k \in [1,n]. \tag{4}$$

The final design samples matrix $\tilde{X}$ in unit hypercube is constructed by concatenating all A, B, and AB matrices vertically, such that,

$$\tilde{X} = \begin{bmatrix} A \\ AB^{(1)} \\ \vdots \\ AB^{(n)} \\ B \end{bmatrix} \in \mathbb{R}^{P(n+2)\times n}. \tag{5}$$

The final design samples X can be obtained from $\tilde{X}$ by column-wise scaling using the upper-and-lower bounds of the corresponding design variables.

The adopted sampling scheme meets our DOE goals. Firstly, the Saltelli sampling scheme is deterministic and reproducible. By generating samples from a Sobol sequence, the method achieves uniform space-filling properties that minimize clustering and sampling bias, leading to balanced datasets ideal for machine-learning model training. Secondly, because the Sobol sequence is nested, doubling the dataset size yields a refined design-space coverage without modifying the existing design points, ensuring that dataset scaling does not require rerunning existing experiments. Moreover, the Saltelli framework naturally supports variance-based sensitivity analysis and uncertainty quantification, thereby enriching the downstream applications of the dataset.

The Saltelli sequence samples X are used exclusively for model training. Because Saltelli sampling enforces a homogeneous distribution, randomly partitioning X to form a test set would disrupt this structure and introduce bias into the remaining training samples. To avoid this issue, we generate a separate holdout test set independent of X with 16 geometries. The holdout test set is generated using LHS sampling via `SALib`'s `sample.latin` function [45]. The resulting unit-hypercube samples are subsequently scaled using the same procedure employed in the Saltelli sampling process described previously.

2. Sampling Results

As a result of the sampling scheme described, one can generate geometry configurations at different levels of density in a nested structure. We took six values of P, in multiples of 2, to generate six levels of datasets, shown in Table 2. Each geometry in the dataset is evaluated at nine AOAs, ranging from 11° to 19° in increments of 1°.

Level	**Base Sequence Length (P)**	**Geometry Configurations**	**Number of AOAs**	**Dataset Size (D)**
1	1	8	9	72
2	2	16	9	144
3	4	32	9	288
4	8	64	9	576
5	16	128	9	1152
6	32	256	9	2304

Table 2 Dataset sizes at six levels.

Together, these configurations constitute of the training geometry set of the dataset. Due to the nested structure of the sampling scheme, the set of geometries in the Level 1 dataset is a strict subset of those in the Level 2 dataset, and this nested structure is preserved across all higher dataset levels. The largest dataset is designated as the Level 6 set. All designs in the Level 6 set are shown in Figure 2(a) in top view, and the corresponding side view at the root in Figure 2 (b). In both figures, the blue edges denote the configurations included in the training geometry set, and the red edges indicate the configurations belonging to the holdout test set.

From Figure 2(a), it is evident that the training set spans a wide design space, ranging from low-aspect-ratio trapezoidal wings (with the largest tip-chord lengths) to delta wings (with the smallest tip-chord lengths). The sampling density is relatively uniform across the design space. Similarly, the holdout test set also exhibits a consistent sampling density, as indicated by the even distribution of red edges across the design space. Similarly, uniform design space coverage for δ can be observed in Figure 2(b).

To further illustrate the sampling density, Figure 3 presents the pair plots of the design variables for samples in the Level 6 set and holdout set. The off-diagonal panels show the scatter plots of paired design variables, while the diagonal panels show the histograms of the sampling density with respect to each design variable. Samples belonging to the training geometry set are plotted in blue, while samples corresponding to the holdout test set are shown in red. As indicated by the paired plots, no specific clustering is observed across any design variable pairs, and within each design variable, a uniform sampling density is achieved across the range of design-variable bounds. When examining the distribution of the holdout test set, the test samples also span the design space evenly. Because the sampling scheme for the training set is nested, the sampling density distribution across training set levels is consistent, and hence the sampling density pair plots for the sparser sampling levels are not shown.

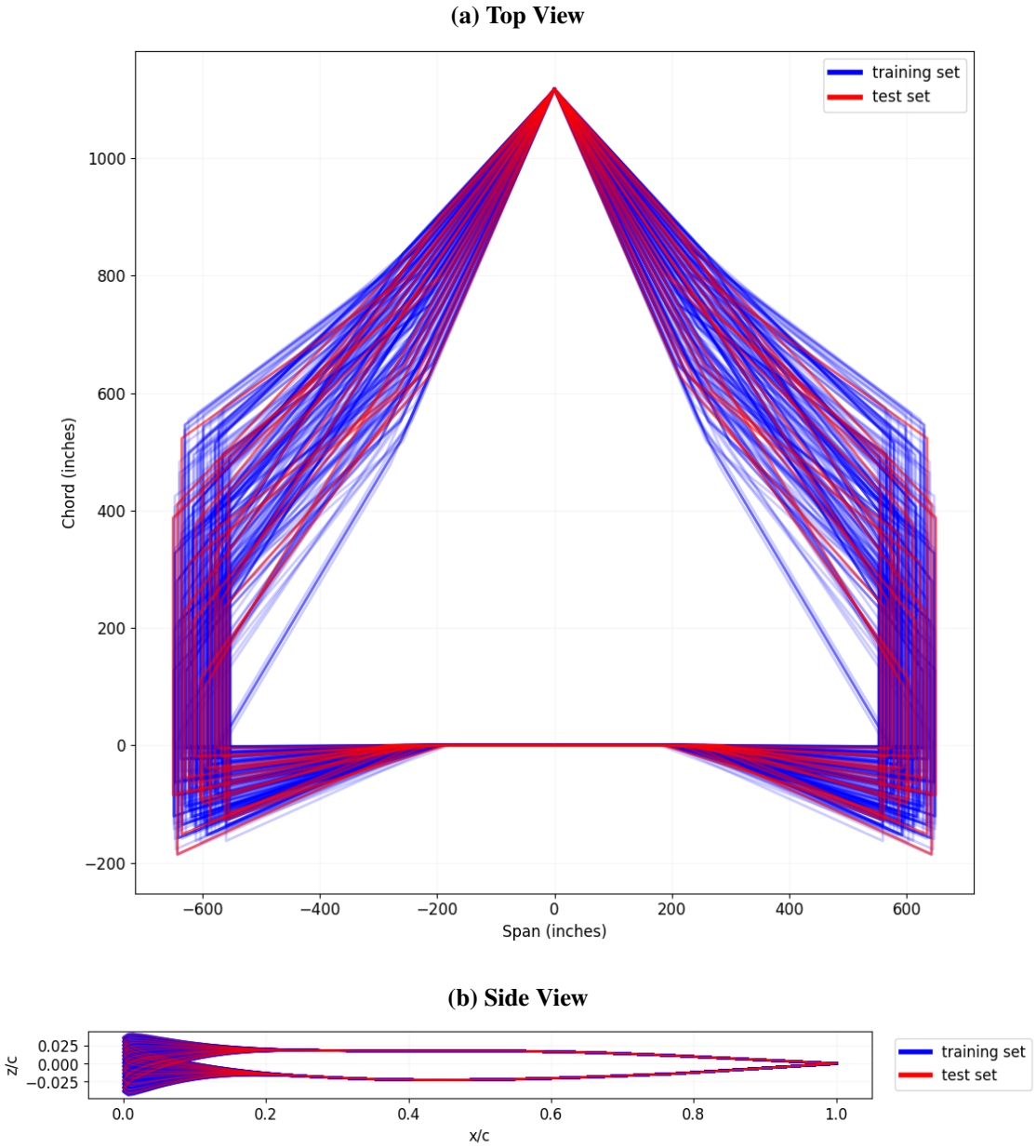


Fig. 2 Illustration of vehicle configurations sampled from the design space. Training samples from the Level 6 set are shown in blue, while samples from the holdout test set are shown in red.

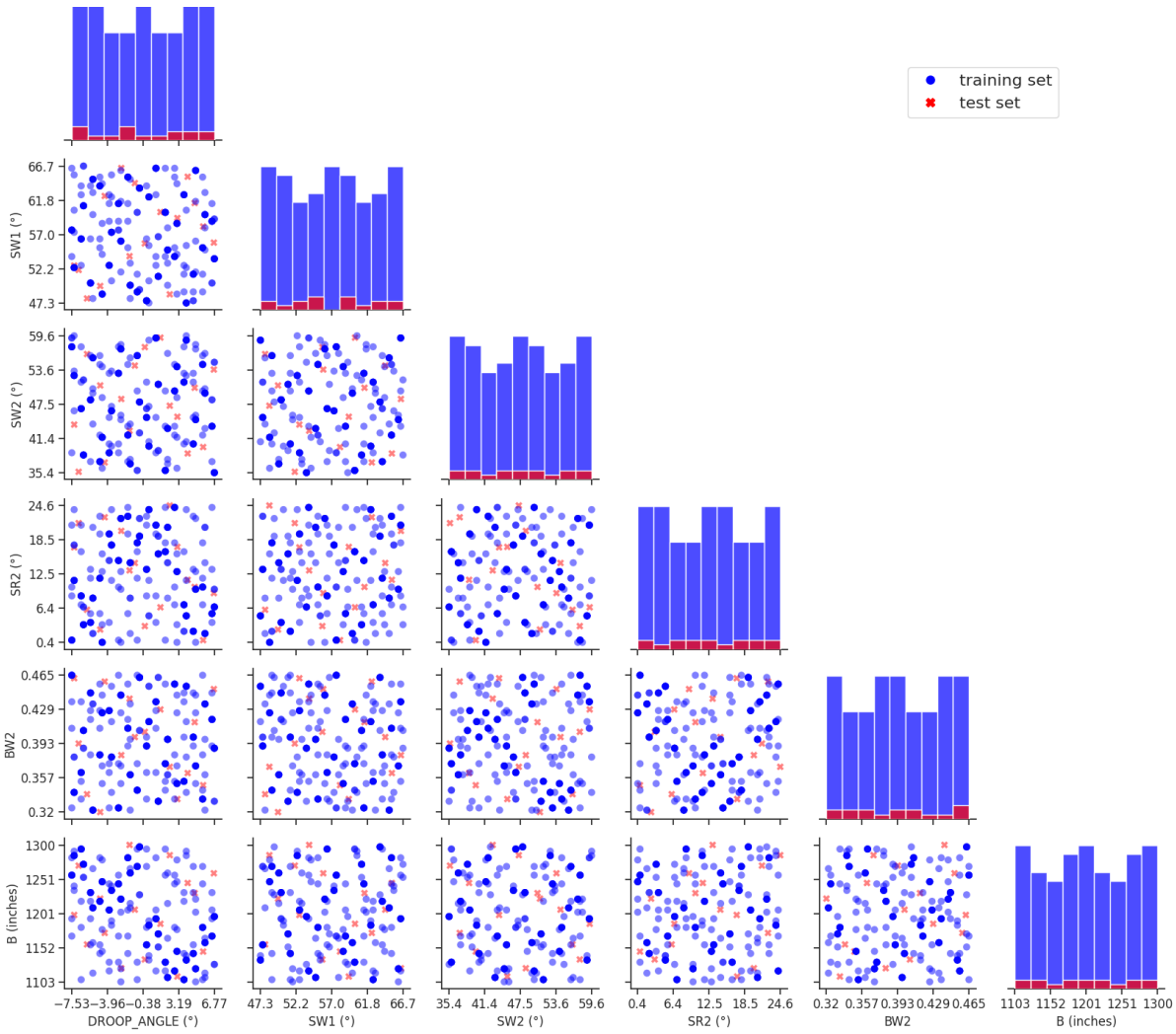


Fig. 3 Design space coverage and design variable distributions of the dataset.

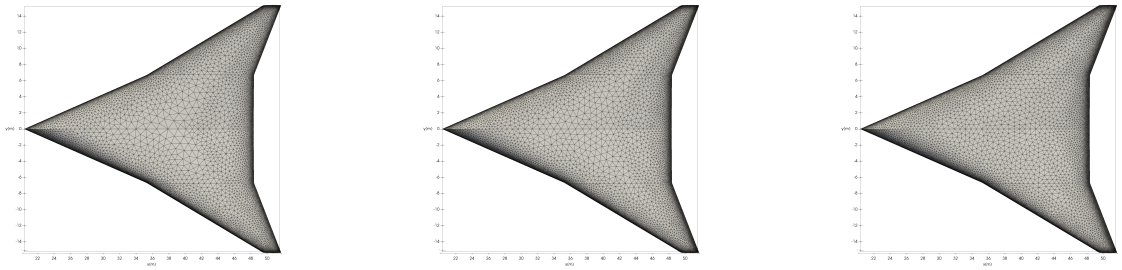
D. Numerical Simulation

1. CFD Meshing

Vehicle geometries are generated from the design variable samples using `SUAVE`'s `write_vsp_mesh` interface to produce the corresponding `OpenVSP` geometry model, as discussed in Section II.A. The CFD mesh generation involves two stages: surface meshes are first produced using `OpenVSP`'s `ComputeCFDMesh` function, and volume meshes are then created in `Pointwise` using its `T-Rex` extrusion function. The `T-Rex` procedure extrudes the surface mesh into the freestream to form an anisotropic boundary layer mesh with controlled growth and resolution suitable for RANS simulations.

Several parameters control the mesh properties of `OpenVSP`'s `ComputeCFDMesh` surface meshes. These parameters include source definitions, the maximum mesh-edge length (enumerator `CFD_MAX_EDGE_LEN`), the maximum edge-growth ratio (`CFD_GROWTH_RATIO`), and 3D rigorous smoothing and size-field limiting using nearest-neighbor matching of adjacent mesh elements (`CFD_LIMIT_GROWTH_FLAG`). For the source definition, we add a trailing-edge source (`WLineSource`) with an influence radius of 0.1 MAC and a mesh-edge-length target of 0.003 MAC, in addition to the default sources at the leading edge and wing tips. The added trailing-edge refinement improved the volume-mesh quality near the trailing edge, and the refinement size is determined based on engineering judgment. The maximum edge-growth ratio is set to 1.2, and 3D rigorous smoothing is enabled. We control the surface-mesh density by adjusting the maximum mesh-edge length. Six levels of mesh refinement are defined, with maximum edge lengths of 0.1, 0.05, 0.0375, 0.025, 0.02, and 0.01 MAC. We select a representative geometry from the dataset to determine a mesh configuration that is applied consistently across the full dataset. These surface meshes are shown in Figure 4. As expected, increasing the mesh-refinement level increases the mesh density. The corresponding surface cell element counts are 1.01×10^5, 1.05×10^5, 1.08×10^5, 1.18×10^5, 1.25×10^5, and 1.82×10^5 from Level 1 to Level 6 respectively.

(a) Level 1 **(b) Level 2** **(c) Level 3**



(d) Level 4 **(e) Level 5** **(f) Level 6**

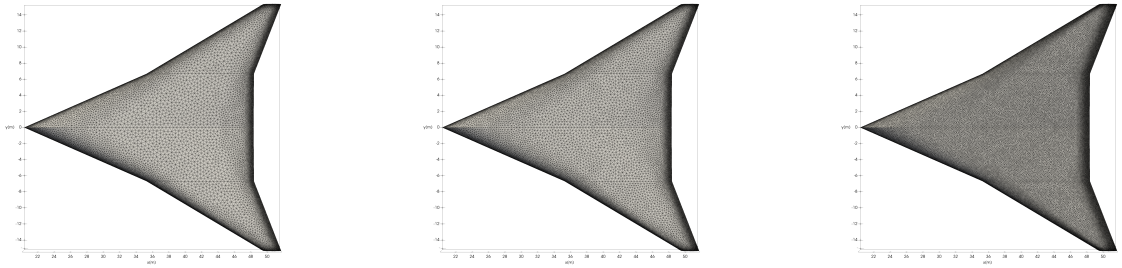


Fig. 4 Surface meshes of a representative geometry used for the mesh sizing study.

The volume mesh is created from the exported surface mesh using a modified implementation of the `VSP2CFD` code [48]. The primary parameters that control the volume-mesh properties are the maximum number of anisotropic `T-Rex` boundary-layer layers (`TRexMaximumLayers`), the first layer thickness (`TRexCondition-setSpacing`), and

the boundary decay rate of the unstructured mesh around wall surfaces (`BoundaryDecay`). For all mesh levels, the boundary decay rate is set to 0.85. The first layer thickness values for the corresponding surface meshes from Level 1 to Level 6 are 6×10^{-5}, 2×10^{-5}, 9×10^{-6}, 7×10^{-6}, 6×10^{-6}, and 6×10^{-6} (m), respectively. The maximum number of anisotropic layers is set to 25, 35, 40, 50, 45, and 50 layers from Level 1 to Level 6. As a result, the total volume cell counts are 5.32, 6.13, 6.57, 6.96, 7.54, and 10.72 million cells, respectively. The volume-mesh sizing study focuses primarily on identifying a proper boundary-layer mesh setup that allows the SA-R turbulence model [34] to capture the boundary-layer flow correctly, thereby yielding accurate aerodynamic and flow predictions. Table 3 summarizes the mesh configurations used in this study.

Level	Surface Elements	Volume Elements	$\bar{y^+}$
1	1.01×10^5	5.32×10^6	6.93
2	1.05×10^5	6.13×10^6	3.50
3	1.08×10^5	6.57×10^6	1.98
4	1.18×10^5	6.96×10^6	1.66
5	1.25×10^5	7.54×10^6	1.35
6	1.82×10^5	1.07×10^7	1.32

Table 3 Mesh statistics across refinement levels.

A mesh convergence study is conducted by running CFD simulations with identical freestream conditions and solver configurations across all mesh levels. We use the representative geometry at Mach 0.3 AOA 13° flow and monitor the convergence of C_D and C_M, as well as the resultant average $\bar{y^+}$. We define the area-averaged y^+ as:

$$\bar{y^+} = \frac{\int_S y^+ dA}{\int_S dA}, \tag{6}$$

and the resulting values for each mesh level are reported in Table 3. For the SA-RC turbulence model without wall functions, we target $\bar{y^+} \approx 1$, which is achieved by selecting mesh level 4 and beyond. The convergence behavior for C_D and C_M is shown in Figure 5. It is observed that the Level 4 mesh with 6.96×10^6 volume cells achieves a difference within 1% of the finest grid used in this study while maintaining a manageable computation cost for large dataset generation. Therefore, this mesh configuration is selected and used for all subsequent configurations' meshing.

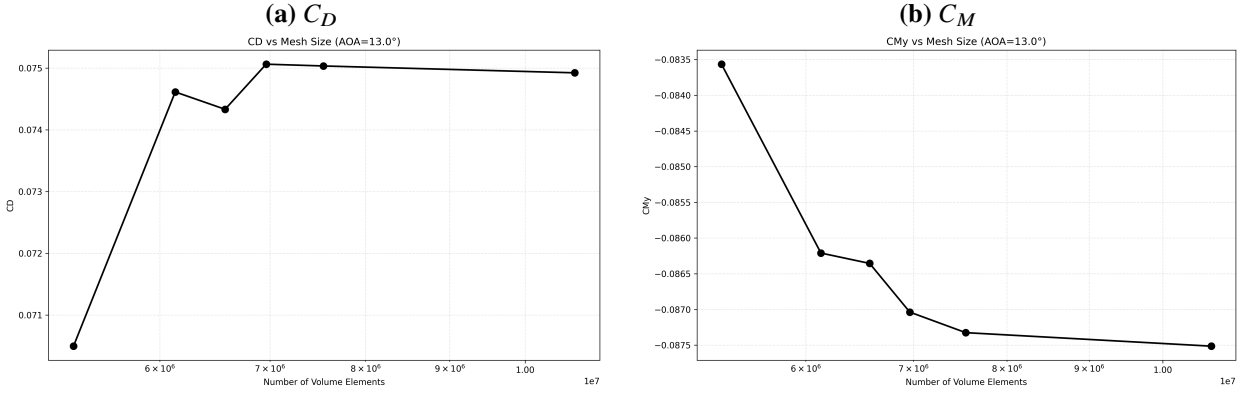


Fig. 5 Mesh sizing study: convergence of C_D and C_M with respect to mesh size.

2. CFD Solver Settings

The fluid domain consists of a freestream far-field with a diameter of 40 MAC being 630 (m), centered at the wing apex, and the wing surface. The far-field boundary is assigned with a far-field boundary condition, while the wing surface is assigned with adiabatic wall (no-slip) condition. Freestream condition is specified as standard air, with gas

constant of 287.06, at 285 (K), and static pressure of 7.18×10^4 (Pa). The air has a constant heat capacity ratio of 1.4. Sutherland viscosity model is used, with a reference viscosity of $1.715 \times 10^{-5}(\mathrm{Ns/m^2})$ [49], and a constant laminar Prandtl number of 0.72 and turbulent Prandtl number of 0.9. The reference Reynolds number is based on average the MAC of 16 (m), yielding $Re_{ref} = 8.04 \times 10^7$. The aerodynamic coefficients are computed using a specific moment origin on the body and reference area defined according to the sizing rules in `SUAVE`. These configuration specific reference values are included in the dataset repository.

Spatial discretization employs the Jameson-Schmidt-Turkel (JST) scheme [50] for the flow equations with Green-Gauss gradient reconstruction, using a JST sensor tuned with coefficients (0.5, 0.02). The turbulence equation is discretized using a scalar upwind scheme without MUSCL reconstruction. Temporal advancement uses implicit Euler for both flow and turbulence, with an adaptive Courant-Friedrichs-Lewy (CFL) number. The linear systems are solved by FGMRES with ILU preconditioning and a residual tolerance of 10^{-7}. The convergence criterion is set to be 10^{-6} for the continuity equation, with an additional convergence criterion for for the drag coefficient C_D requiring changes below 10^{-5} over 100 Cauchy iterations. These convergence criteria are applied consistently throughout all runs in the dataset.

3. CFD Results and Discussion

CFD runs are conducted using the Stanford Research Computing Center (SRCC) facility with a 32-core 2.4 GHz partition. The total computational cost of the dataset is 310, 835 CPU-hours for 2, 448 snapshots. For illustration of the variety of flow conditions represented in the dataset, Figure 6 shows three representative configurations (call signs as subtitles) at AOA of 15°. These geometries are selected from the holdout test set, spanning the platform types ranging from trapezoid wings, cropped Delta wings, to double-delta wings. The wing surfaces are colored by C_P, and Q-criterion iso-surfaces of 100 are added to illustrate the vortex sheets. The variation of geometries leads to drastically different surface pressure distributions and primary vortex structures, enriching the geometric and flow features for surrogate modeling of flow prediction of the double-delta wing family.

(a) 050101-7LK **(b) 09190107DL** **(c) 171801-1NW**

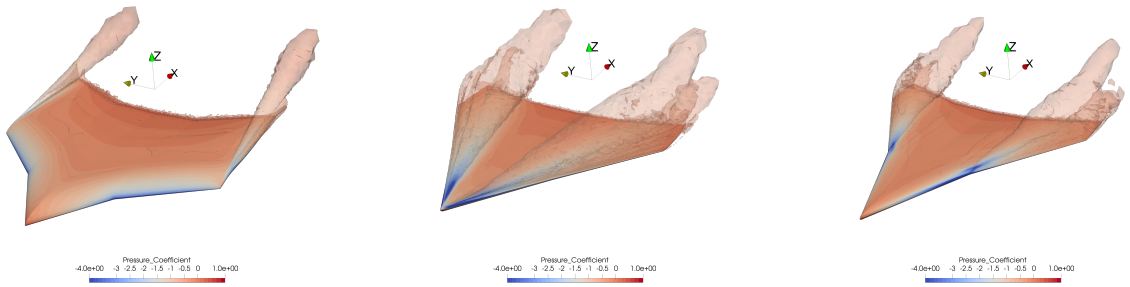


Fig. 6 Pressure coefficient (C_P) visualizations and Q-criterion iso-surfaces at $Q = 100$ for different vehicle configurations, illustrating variations in geometry and flow states.

Among the 2, 448 CFD simulations, 57 runs failed to converge to the specified residual target using the numerical scheme described above. These runs correspond primarily to highly cambered geometries at higher AOAs above 16°. The convergence failure arises from the unsteady nature of critically separated flow, for which the selected physics models and numerical scheme cannot recover a steady solution at the residual target. We believe it is more valuable to provide solutions with consistently converged results, even at a lower fidelity, rather than removing these snapshots. As a result, we lowered the order of the convective scheme and adopted AUSM [51] for these runs. The configuration files contained in the dataset annotate which convective scheme is used for each run. Future extensions of the dataset may consider using higher-fidelity methods, such as detached eddy simulation, for these runs.

E. Data Availability

The dataset is hosted on our Huggingface repository at https://huggingface.co/datasets/yirens/double-delta-aero under GNU Lesser General Public License v2.1. The repository includes all CFD results, including the `SU2` configuration files, surface and volume solutions in `vtk` format, convergence history files, and integrated aerodynamic loading

summary files. The VLM solutions are provided in `json` format. All files are organized and indexed by vehicle call sign for ease of access.

IV. Data Size Scaling Relationship

Neural scaling laws quantitatively describe how model performance improves with increasing training scale. The training scale comprises three major components: data scaling (how performance changes with increasing training dataset size), model scaling (how performance changes with increasing model size), and computation scaling (how model performance changes with increasing computational budget). Foundational work was carried out by Hestness et al. Rosenfeld et al., Kaplan et al., and Hoffmann et al., focusing on training large language models [25, 52–54]. Together, the studies show that model performance under a given computational budget C, when measured by an appropriate loss metric, follows a power-law relationship with training dataset size and model size:

$$\epsilon = a_1 N^{-\alpha} + a_2 D^{-\beta} + \epsilon_0, \tag{7}$$

where ϵ is the test loss, N is the model size, D is the dataset size, and ϵ_0 is the irreducible error. Coefficients a_1, a_2, α, β, and ϵ_0 are fitting parameters. Such formulation is applicable to GNNs, as indicated by previous study [26].

We define ϵ as the mean squared error (MSE) evaluated on the holdout test set. Recall that `MF-VortexNet` learns a mapping $\mathcal{F}$ that takes a low-fidelity pressure–stream-condition–geometry state, represented as a graph G, and produces a predicted pressure distribution over the lattice, C_p^{pred}, that approximates the ground-truth pressure distribution C_p^{gt} obtained from high-fidelity simulations. The test loss is then defined as

$$\epsilon = \text{MSE}(C_p^{\text{gt}}, C_p^{\text{pred}}). \tag{8}$$

The remainder of this section outlines the procedure used to identify the scaling-law coefficients based on the test error ϵ, with the results interpreted as preliminary.

A. Experiment Setups

The empirical scaling experiment aims to estimate the exponent coefficients in Equation 7, enabling us to quantify how prediction accuracy scales with training data size and to determine the data requirements necessary to achieve a target accuracy. The experiment comprises three components: the training data, the model, and the training configurations.

1. Data

We define the training data scaling under investigation as the increase in the number of snapshots resulting from higher sampling density over the a defined geometric design space. As summarized in Table 2, using Saltelli sampling, the geometric configurations in the training set (described in Section III) are subdivided into six levels. From Level 1 to Level 6, the training set includes a total of 8, 16, 32, 64, 128, and 256 geometric configurations, respectively. For each level, flow snapshots at five AOAs are selected at 11°, 13°, 15°, 17°, and 19°. Together, our training data scaling involves D values of 40, 80, 160, 320, 640, and 1280 graphs, respectively. Each graph provides a multi-fidelity representation of a geometric configuration and its associated flow state.

The decision to include only five AOAs per geometry, rather than all available AOAs in the dataset, is motivated by the need to balance dataset size and model capacity under tight computational budget. Because the primary objective of the data scaling study is to determine the geometric sampling density required to train `MF-VortexNet`, we prioritize scaling the number of geometries rather than the number of flow snapshots per geometry. As discussed by Hoffmann et al., dataset size, model size, and computational budget should be scaled proportionally [54]. Subsampling the available AOAs therefore enables training with smaller models at lower computational cost, while preserving the qualitative empirical scaling trends.

An additional consideration concerns how to define the dataset size D for a collection of graphs. Prior studies across different domains have adopted different definitions, including the volume of digital information measured in terabytes [55], the number of individual graphs [56], and the total number of edges [26]. Consequently, there is currently no standardized definition of D in graph-based setting, and its interpretation remains domain-specific. In this work, we define D as the total number of graphs in the dataset. This definition is aligned with the downstream design application, where designers seek to understand how densely the design space needs to be sampled and how many high-fidelity

simulations are required to train a surrogate model with sufficient accuracy. Such a definition directly links to the number of simulation runs required and is thus more convenient for budget planning.

Besides the training set, we constructed the test set using the sixteen holdout test geometries, as described in Section III. To maintain the consistency with the training set in flow snapshot selection, we include include snapshots at AOAs of 11°, 13°, 15°, 17°, and 19°. In total, the test set comprises 80 graphs. The test set is relatively small. However, considering the even geometric distribution shown in Figure 3, we believe the test set contains adequate variability representative of the training set and is suitable for use in test loss evaluation.

2. Model

Prior work has shown that performance scaling depends critically on the relationship between dataset size and model capacity. In the under-parameterized or over-parameterized regimes, corresponding to a disproportionately small or large ratios of N/D, the test error no longer exhibits a power-law decay with respect to D, a phenomenon referred to as model scaling collapse [26, 54]. To avoid inconclusive results derived from a single fixed-capacity model, we repeat the scaling experiment using four models with model sizes ranging from 0.10 million to 2.41 million trainable parameters.

The `MF-VortexNet` architecture has three hyper-parameters that are most relevant to model size [10], including the width of the nodal encoding for hidden layers (`hidden_feature`), the number of heads for multi-head attention (`heads`), and the total depth of the message passing blocks (`hops`). The `hops` parameter is related to the extent of nodal information propagation, and for the given 30×32 lattice, we determined that a `hops` of 10 is adequate for the information to propagate across all nodes in a graph. This hyper-parameter is therefore held constant across models. To change the model size, we alter `heads` and `hidden_feature`. Specifically, increasing `heads` and `hidden_feature` increases the model size while widening the model. The resulting models created, along with their numbers of trainable parameter size (N), are presented in Table 4.

Level	**`hidden_feature`**	**`heads`**	**Model Size (N)**
Mini	13	5	0.103×10^6
Small	27	5	0.538×10^6
Medium	37	5	1.144×10^6
Large	45	6	2.411×10^6

Table 4 Model sizes at four levels.

Once the model size is defined, additional hyper-parameters related to regularization and training must be specified [10]. Assigning a single fixed set of hyper-parameters across all models is not feasible, as models of different sizes require different training dynamics and regularization strategies to achieve convergence and target performance accuracy. Instead, we promote cross-model comparability by adopting a consistent hyper-parameter selection procedure that yields "equally optimal" configurations for each model.

For each model size, we employ `Optuna` [57] to optimize key training and regularization hyper-parameters, including the learning rate and weight decay of the ADAM optimizer, as well as the dropout ratio and skip-connection bandwidth (`Alpha`). Hyper-parameter optimization is performed using a subset of the Level 6 training set, constructed by randomly sampling 2048 snapshots, thereby ensuring that no information from the holdout test set is leaked during model selection. This dataset is further randomly split into an 80%-20% training-validation partition. For each model, the training configuration that achieves the lowest validation MSE is adopted as the training hyper-parameters.

To ensure fair comparison across model capacities, an identical computational budget, as discussed in the following section, is allocated to all models. We conduct a maximum of 35 training runs during hyper-parameter optimization for each model, with the first ten trials used as "warm start" evaluations for `Optuna`. All other `Optuna` optimization settings are left at their default values.

The MSE values of the optimized models are reported in Table 5. Despite differences in model sizes and training configurations, comparable test MSE values are achieved. The remaining hyper-parameters are reported to ensure reproducibility. These hyper-parameters are held fixed for each model across all data scaling runs.

Level	Test MSE	Learning Rate	Decay	Dropout	Alpha
Mini	5.97×10^{-2}	1.14×10^{-3}	2.81×10^{-6}	5.57×10^{-2}	4.50×10^{-1}
Small	5.66×10^{-2}	6.59×10^{-3}	1.54×10^{-6}	1.22×10^{-1}	2.38×10^{-1}
Medium	5.21×10^{-2}	1.60×10^{-3}	1.98×10^{-6}	5.19×10^{-2}	4.10×10^{-1}
Large	5.88×10^{-2}	6.02×10^{-4}	3.28×10^{-6}	1.82×10^{-1}	2.77×10^{-1}

Table 5 Additional training hyper-parameters after optimization for all models.

3. Training

To fit the neural scaling law coefficients in Equation 7, we conduct a controlled data-scaling experiment for each model listed in Table 4. The training-dataset size D is systematically varied, as discussed in Section IV.A.1, while keeping the training procedure fixed. This procedure is designed to isolate the effect of data scaling from variations in model capacity or computational cost.

When performing the D scaling using datasets at six levels, as discussed in Section IV.A.1, and across four model sizes N, careful design of the training scheme is needed such that, as a result of the experiment, the fitted β represents the training efficiency gain resulting from increased geometric sampling density, rather than being a consequence of increased computation. As an experimental design choice, we fix the maximum computational budget C^* available for all training runs across all D and N. C^* is defined as the maximum number of weights update steps available for each training fold, and is set to 2000. Four folds are used to train the model; in each fold, a 75%–25% training-to-validation data partitioning is used. Adopting k-fold cross-validation reduces the risk of model overfitting and is a critical component of weight scheduling for physics-informed loss integration [10]. In addition to fixing C^*, an early stopping patience of 50 epochs is used across training runs to prevent model overfitting. The early stopping mechanism terminates the training of the current fold when no improvement in validation loss is observed over the preceding 50 epochs. After examining the validation loss histories across epochs for all runs, we did not observe typical overfitting behavior, such as validation loss increases toward the end of training, except for the large model trained on the Level 1 dataset, which, due to the limited amount of validation data, is more prone to overfitting, supporting the appropriateness of the chosen early stopping patience.

Setting a fixed maximum number of weight-update iterations introduces an additional challenge for dataset-size scaling. In standard mini-batch gradient descent, each batch corresponds to a single weight-update step; for a training dataset of size D_t and batch size B, a total of D_t/B weight updates are performed per epoch. Since B is constrained by hardware memory and is fixed at 128, the total number of weight updates scales with D_t, introducing an inconsistency when isolating the effect of dataset size on model performance. Moreover, prior work suggests that the optimal computational budget C does not necessarily scale linearly with D [54], and the scope of the present study is insufficient to identify the optimal scaling relationship between C and D. Consequently, during the experiment we found the model hyper-parameters, as discussed in Section IV.A.2, are sensitive to C, such that designing a C allocation scheme across model sizes and data sizes for calibrating the performance becomes difficult. An imprudent C scheduling may result in biased power-law observations.

To address this issue and isolate error scaling with respect to D alone, we fix the number of weight-update steps across dataset sizes by adopting a stochastic subsampling scheme. Specifically, at each epoch, we randomly draw B samples from the training set and perform a full-batch gradient update. Because of the stochastic nature of this procedure, and given a sufficiently large number of epochs, the model is exposed to the entire training dataset in expectation, while the total computational budget C remains constant across different dataset sizes. As a result, for each fixed model size, the observed scaling behavior can be attributed solely to changes in sample distribution introduced by D scaling. Since the same number of flow conditions are selected at each level of the geometry set, this scaling behavior directly reflects gains in prediction accuracy arising from increased sampling density in the design space. The resultant ϵ versus D scaling results are conceptually similar to the iso-FLOP experiments reported by Hoffmann et al. [54].

The remaining training setup follows Shen et al. [10]. For each model training at a particular model size N and dataset size D, we repeat the training six times under randomized initial seeds for initial model weights and dataset partitioning. Trial-level MSEs are aggregated to report per-level mean, standard deviation, minimum, and maximum values to improve the robustness of the scaling exponent estimation. Least-squares fitting using `SciPy`'s `curve_fit` function is used to fit the aggregated mean MSEs to the scaling law coefficients in Equation 7. All model trainings are executed on a 40GB NVIDIA A100 GPU.

B. Results and Discussion

The mean test MSE, evaluated on the holdout test set and averaged across all trails at each D, is plotted in Figure 7. Different data series represent scaling results from different models, from large to small. Overall, the test MSE decreases as D increases, although the rate of convergence varies across models. From the learning curves, three distinct learning regions can be identified. We highlight these regions by light red, white, and light green bands along the D axis. Such zoning is consistent with the learning curve behavior described by Hestness et al. and Rosenfeld et al. [52, 53].

When the dataset is small (light red region), the learning curves exhibit behavior characteristic of the "small data region"[52], in which model performance is limited by insufficient training data provided. Consequently, the model performs a best-guess prediction. For models investigated in this study, this region corresponds to $D \leq 80$. As D increases beyond the "small data region", a "power-law region" [52] emerges, such that increasing D in this region helps the model to improve its prediction quality for generalization tasks, such as predictions in the holdout test set. Under the current model and training configuration, this region spans $80 \leq D \leq 320$, corresponding to geometry sets of 16 to 64 configurations.

For larger data sizes, at $D \geq 320$, an "asymptotic error region" is observed. We intentionally distinguish the terminology of this region from the "irreducible error region" described by Hestness et al. [52], as the error floor observed in Figure 7 should not be interpreted as an inherent limitation of the dataset or as insufficient model expressivity. Instead, it arises from training under a fixed computational budget C^*, as discussed in Section IV.A.3. To achieve fixed C^* across different values of Ds, subsampling is used. Under the training setup of a batch size of 128, training data for $D \geq 170$ cannot fit into the full-batch gradient descent, and thus the training data, when viewed as an entire set, are underutilized in each weight update step. Although all graphs in the training set are likely to be used during training, each graph in larger datasets is less exposed to the optimizer, and hence the resulting models are less optimally trained. This under-training effect explains why the test MSE for models trained at $D = 1280$ tends to be higher than those trained at $D = 640$. These trends are consistent with the iso-FLOP curves reported in previous literature [54], and this region should not be used for power-law fitting as the model training is bottlenecked by C^*.

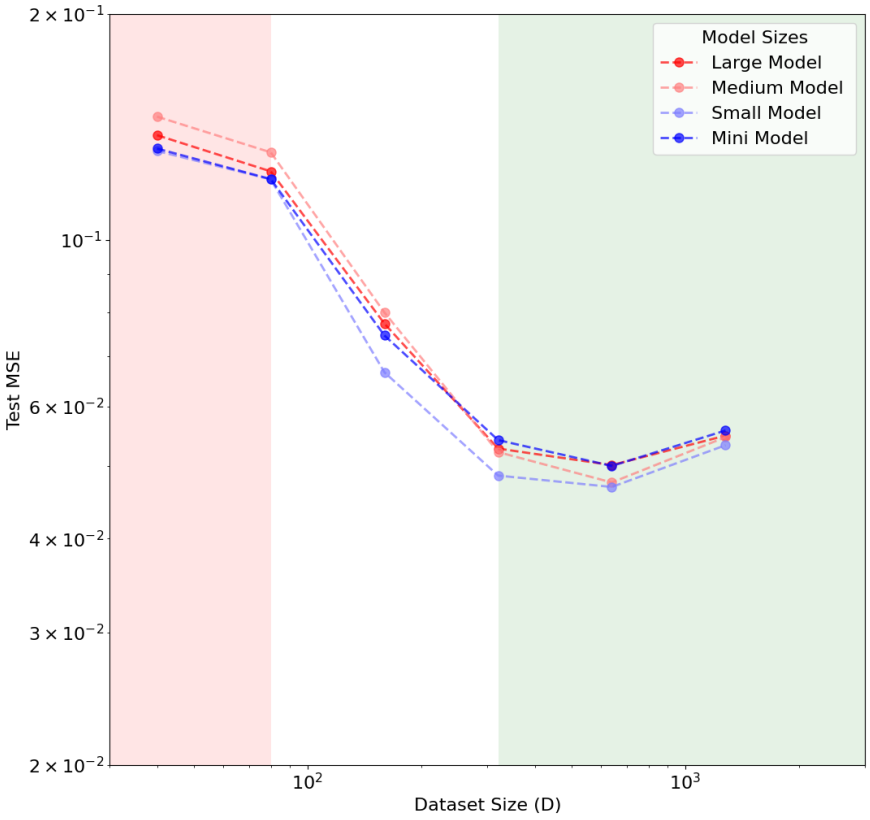


Fig. 7 Scaling behavior of test MSE as a function of the training set size D for models of different sizes, where D denotes the number of distinct geometry-flow snapshots represented as graphs in the dataset.

Among these regions, the power-law region provides information on data-size scaling. Figure 8 presents the test MSE as a function of D, together with the mean value, variability indicated by boxes spanning plus and minus one standard deviation, and the minimum and maximum values shown by whiskers in a box-plot style. The mean values are identical to those presented in Figure 7. It is observed that the larger models tends to have smaller coefficient of variation in the power-law region, indicating a more consistent performance across repeated trials.

Within the power-law region, Equation 7 can be further reduced to,

$$\epsilon' = a_2 D^{-\beta}, \tag{9}$$

by absorbing the constant contributions from model size and irreducible error into the test loss, we focus on quantifying the scaling exponent β. To do so, we use `SciPy`'s `curve_fit` function to perform a non-linear least squares fit of the data. The mean MSE is treated as the dependent variable, and D is treated as the independent variable, with the corresponding standard deviations used to represent the relative uncertainty of the MSE measurements. The resulting power-law fits are shown as dashed lines in Figure 8(a)–(d).

From the power-law fitting, we estimate the scaling exponent β to be 0.5484, 0.6220, 0.6654, and 0.6128 for the Mini, Small, Medium, and Large models, respectively. The corresponding coefficient a_2 are 1.2617, 1.7354, 2.4044, and 1.7778, respectively. The corresponding R-squared values are 0.9772, 0.9652, 0.9985, and 0.9965 respectively, indicating a high-quality fits for β estimation.

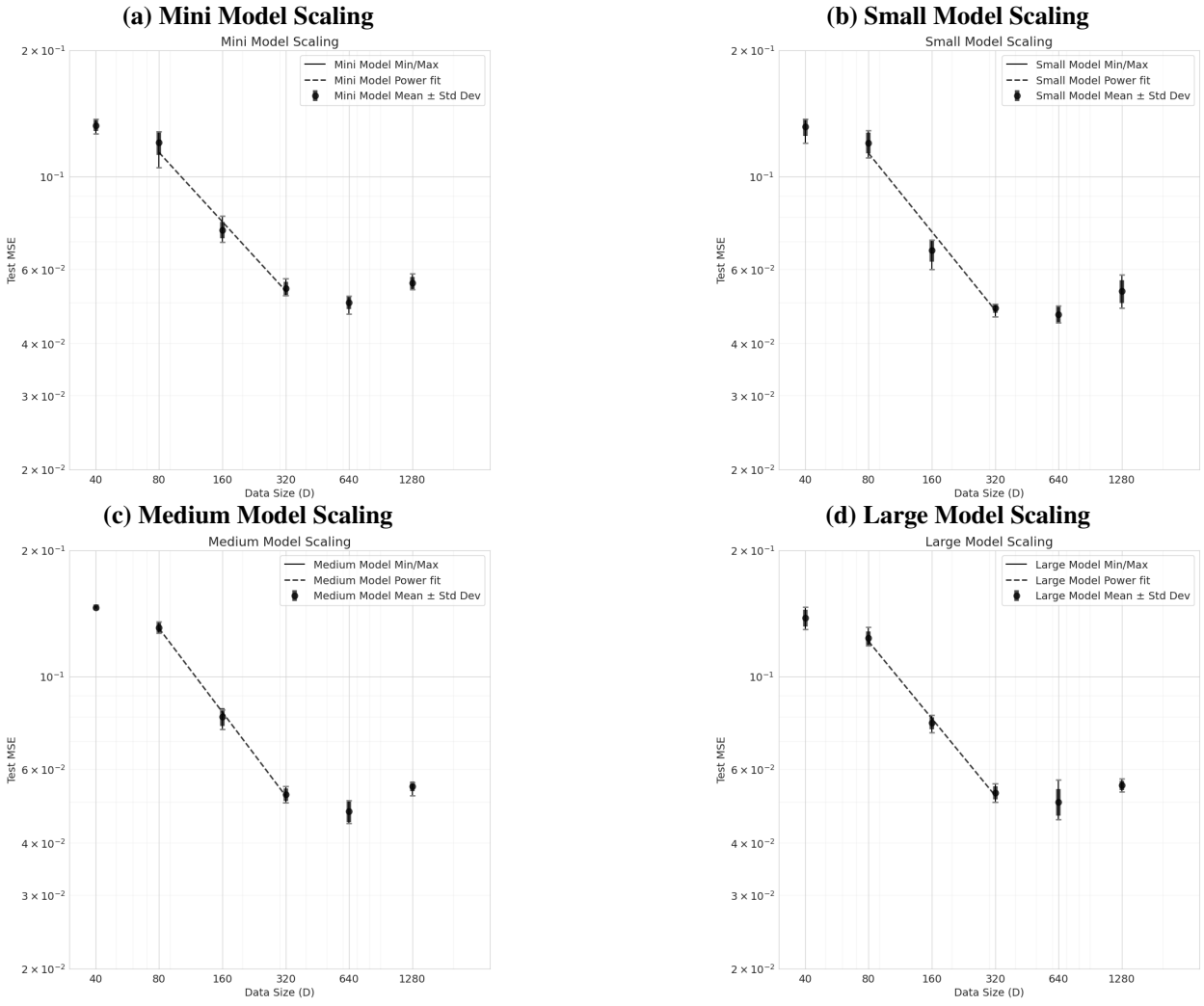


Fig. 8 Test MSE scaling with D for four models arranged from (a) to (d): Mini, Small, Medium, and Large, along with their log-log linear fit for the power-law region.

These fitted power laws can then be used to answer the question of how much data is required to achieve a target prediction accuracy under the proposed training procedure and computational budget. Assuming that the minimum MSEs obtained from hyper-parameter optimization (Table 5) are reasonable estimates of models' best performance under substantially larger dataset and are served as the loss target, we compute the optimal dataset size D^* at the computational budget C^* by inverting Equation 9. The resulting values of D^* are 261, 245, 317, and 261 for the Mini, Small, Medium, and Large models, respectively. These values correspond to geometry sets of 52, 49, 63, and 52 geometries, all of which are slightly smaller than the Level 4 dataset listed in Table 2.

The size of the geometry set can be further mapped to the corresponding sampling density in the unit hypercube. To this end, we first fit a power law relating the average nearest-neighbor distance h, defined as the average Euclidean distance from each design sample to its nearest neighbor, to the number of geometry samples G contained in a dataset, using the data in Table 2. This fit yields $h = 0.62G^{-0.15}$, from which the estimated D^* values correspond to h in the range of 0.333 to 0.346. The corresponding one-dimensional optimal sampling density can then be estimated for space-filling sampling algorithms by accounting for dimensional scaling. Based on this analysis, selecting approximately eight sampling points per dimension is sufficient to achieve optimal surrogate model performance under the allocated computational budget C^*.

The data requirement for training `MF-VortexNet` is considerably lower than those of other machine-learning surrogate models capable of field prediction [15, 16, 18, 21, 22]. Such models typically require hundreds of geometries to achieve geometric generalizability, along with thousands of flow simulation snapshots to achieve field prediction accuracy. We believe the efficiency in data scaling roots in `MF-VortexNet`'s design in operating effectively in small-data use cases. By incorporating physics-informed loss functions, scheduled training strategies, and relatively compact model architectures, prior studies [10, 24] have demonstrated that strong interpolation capability within the design space can be achieved using only 15 to 27 geometries with 189 to 600 flow snapshots. However, because the model remains relatively small compared to large-capacity architectures [18], its representational capacity is inherently limited. As a result, this model size and training scheme constraint may prevent the surrogate from fully exploiting the benefits of substantially larger datasets.

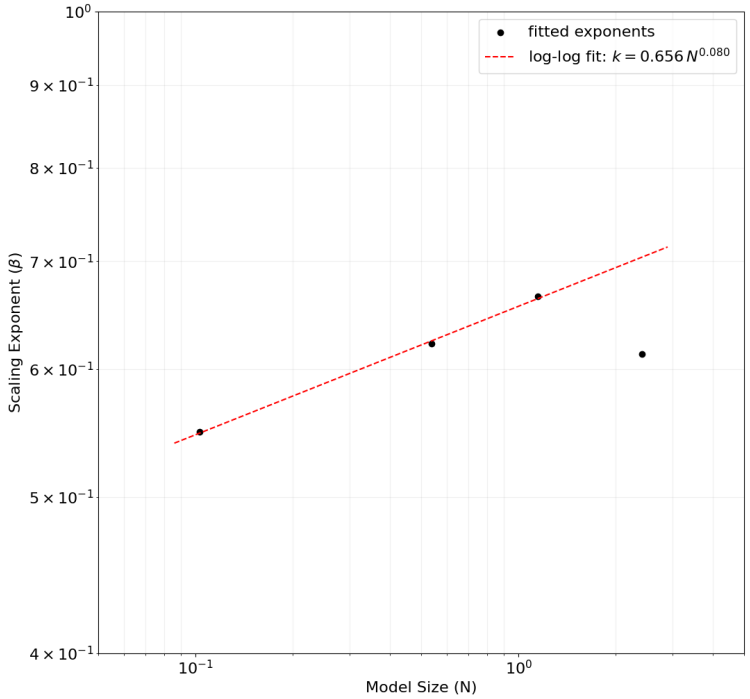


Fig. 9 Exponent (β) scaling with model size (N).

Finally, we assess how the fitted scaling exponent varies with model size. Figure 9 shows the scaling relationship. We observe an approximately linear correlation between β and N on a log-log scale when the model size is not excessively large. The Large model, however, deviates from this trend. The exact cause of this deviation requires further

investigation, but one plausible explanation is that the model capacity exceeds what can be effectively optimized under the available computational budget. As shown in Table 5, the Large model employs a learning rate that is an order of magnitude smaller than those of the smaller models and achieves a higher test MSE than the Medium model, which contradicts typical expectations from model-size scaling [25, 54]. We therefore interpret the selected hyper-parameters, although optimized using the same systematic procedure as for the other models, are compromises between training stability and performance under a fixed computational budget. It is likely that the Large model's capability exceeds what is permitted by the optimizer to drive convergence beyond local minimum under C^*, and as a result the model's representational capability is optimization-limited. Future work will extend the present data-scaling study to model trainings with higher computational budgets to better assess scaling behavior for larger models.

If the computational budget limitation is indeed the cause of the lower fitted β for the Large model, we would expect that increasing the computational budget C^* would lead to an increase in the estimation of β's magnitude for the Large model, conditioned on proper adjustments of training hyper-parameters. Focusing on the overall trend in Figure 9, we conjecture that larger models tend to exhibit larger magnitudes of β, indicating faster convergence and more efficient utilization of training data, until the allocated computational budget becomes the limiting factor. Such behavior of larger models are more efficient to train [53] under limited computational budget is supported by other studies in the language modeling domain [58, 59], and this is due to the greater capacity of larger models to learn task-adaptive representations that extract more features from each data sample. This conjecture suggests that even for domain-specific surrogate models, such as `MF-VortexNet`, there are benefits of constructing larger and wider models: increased model capacity not only enhances field representation and generalization across flow and geometric configurations, but also improves data utilization efficiency during training. As a consequence, larger models may require fewer training samples to achieve a target accuracy, potentially reducing the cost of dataset construction.

However, larger models also demand higher computational budgets for training. In practice, particularly during the conceptual design phase, computational resources are often pre-allocated. The optimal allocation of computational budget between dataset construction and model training when using domain-specific surrogate models therefore depends on the relative costs of generating CFD datasets and training surrogate models, and should be determined by jointly considering these costs together with the training-efficiency gains presented by model size scaling.

V. Conclusion and Future Work

This study addresses two gaps in data-driven surrogate modeling for aerodynamic design: the limited presence of open-source, multi-fidelity datasets for flow simulations with complex physics over relatively simple geometries, and the lack of quantification of the empirical scaling laws for field-predicting GNN-based surrogate models.

A primary contribution of this work is the release of an open-source, multi-fidelity aerodynamic dataset for a parametric family of double-delta wings. To support data-driven modeling of nonlinear, vortex-dominated flows, we generated 272 unique geometries and evaluated each configuration at $Ma = 0.3$ and angles of attack ranging from 11° to 19° using both VLM and RANS CFD solvers. The design of experiments (DOE) is carefully designed to enable future expansion of the dataset and to support variance-based sensitivities analyses. This dataset is publicly available and intended to serve as a benchmark for multi-fidelity surrogate modeling research.

Using this dataset, we conduct a preliminary empirical scaling study of the `MF-VortexNet` surrogate model. The scaling analysis is performed on six datasets, containing between 8 and 256 double-delta wing designs distributed in a hierarchical manner through sequential refinement of a design space's sampling density. By selecting snapshots at specific free-stream conditions, we assembled six training datasets with sizes D ranging from 40 to 1280 graphs. Four models, with sizes ranging from 0.1 to 2.4 million trainable parameters, are trained under a fixed computational budget of at most 8000 weight update steps, subject to early stopping criteria. Collectively, we estimate that the test performance ϵ', evaluated using the test MSE on the holdout test set, scales with D as $\epsilon' \propto D^{-0.6122}$. The observed scaling dependency is strong, indicating efficient utilization of the training data. Leveraging this scaling exponent, and benchmarking against the minimal test MSE achieved using a larger training dataset of size 2048, we estimate that the optimal geometric sampling distance for `MF-VortexNet` under the current computational budget is approximately 0.34 in a six-dimensional design space, which corresponds to a sampling density of eight samples along each dimension using space-filling sampling techniques. The relatively sparse sampling requirement supports the design intent of `MF-VortexNet` as an efficient multi-fidelity surrogate model for domain-specific applications.

By comparing the fitted data size scaling exponents across differently sized models trained under a fixed computational budget, we observe a log-log linear correlation between the model size and data sizing exponent, although the largest model trained in the current study does not follow this trend. We speculate that this anomaly in the largest model is

caused by insufficient computational budget, resulting in a set of suboptimal hyper-parameters for training the model. We conjecture that, when sufficient computational resources are allocated, the magnitude of data scaling exponent increases with larger model size, thereby allowing more efficient utilization of the dataset for larger model. Following this conjecture, larger surrogate model could be trained with less training data, yielding lower cost for offline dataset generation, at the expense of increased model training computation requirements. Consequently, there exists a trade-off between the computational budget allocated to dataset generation and that allocated to model training. Further study is required to systematically vary the computational budget for surrogate model training and to evaluate additional model sizes, in order to obtain more statistically robust estimates of the power-law exponents and to better characterize this trade-off.

In the future, several extensions of the existing work can be made. Firstly, downstream applications, such as sensitivity assessments of surrogate model performance with respect to geometric design variables, can be performed leveraging the DOE used for the dataset. In addition, as approximately 2.3% of the snapshots in the dataset require higher-fidelity CFD solver to achieve convergence, incorporating higher fidelity solutions into the dataset may further enrich its fidelity levels. The preliminary assessment of the data-size scaling relationship conducted in this study indicates that the scaling-law analysis is incomplete without systematically varying the computational budget along with data and model size scaling. In future work, we aim to jointly scale the computational budget, model size, and dataset size to identify optimal resource-allocation strategies.

VI. Acknowledgment

This work was supported by NASA, Award 80NSSC19K1661, under the Commercial Supersonics Technology (CST) program, Supersonic Configurations at Low Speeds (SCALOS), with Sarah Langston as the NASA technical grant monitor. The authors extend their gratitude to Kuang-Ying Ting, Eli Livne, and Chester Nelson for their insightful discussions on the CST design space, and Aurelien Ghiglino for his discussion on the scaling experiments. Additionally, the authors would like to thank Stanford Research Computing Center for providing computational resources on the Sherlock cluster, as well as the Google Cloud Research Credit Program for granting access to GPU resources on the Google Cloud Platform.

References

[1] Wissink, A. M., Liu, M., House, A. M., Kaminsky, A., Simin, A., and Ranjan, R., "Multi-Fidelity Surrogate Modeling for Aerodynamics Using Sage," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0039, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0039.

[2] Peherstorfer, B., Willcox, K., and Gunzburger, M., "Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization," *SIAM Review*, Vol. 60, No. 3, 2018, pp. 550–591. https://doi.org/10.1137/16M1082469, URL https://doi.org/10.1137/16M1082469.

[3] Wassing, S., Langer, S., and Bekemeyer, P., "Adopting Computational Fluid Dynamics Concepts for Physics-Informed Neural Networks," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0269, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0269.

[4] Abras, J., Hosseinverdi, S., Jude, D. P., Sitaraman, J., and Hariharan, N. S., "Machine Learning Based Surrogate Model Development Using GPU Driven Computational Simulations," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0040, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0040.

[5] Healy, R. C., Anusonti-Inthra, P., and Beals, N., "Neural Network Modeling of Airfoil Aerodynamics for Rotorcraft Analysis," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0270, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0270.

[6] Anand, A., Safdar, M. M., Marepally, K., and Baeder, J. D., "A Comprehensive Review of Neural Network Training Approaches for Airfoil Design and Optimization," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0271, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0271.

[7] Kilic, D., and Nikbay, M., "A Comparative Study on Multi-Fidelity Machine Learning Modeling for Aerospace Problems," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0272, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0272.

[8] Anusonti-Inthra, P., and Healy, R. C., "Development of Multi-Fidelity Surrogate Models for Rotor Performance Prediction," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0490, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0490.

[9] Lee, H., Seshadri, P., and Rauleder, J., "Finite Wing Pressure and Lift Coefficient Prediction Using Bayesian Inference on Pressure Data," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0493, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0493.

[10] Shen, Y., Needels, J. T., and Alonso, J. J., "VortexNet: A Graph Neural Network-Based Multi-Fidelity Surrogate Model for Field Predictions," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0494, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0494.

[11] Zhao, H., Gao, Z.-H., and Xia, L., "Efficient aerodynamic analysis and optimization under uncertainty using multi-fidelity polynomial chaos-Kriging surrogate model," *Computers Fluids*, Vol. 246, 2022, p. 105643. https://doi.org/https://doi.org/10.1016/j.compfluid.2022.105643, URL https://www.sciencedirect.com/science/article/pii/S0045793022002390.

[12] Mrosek, M., Othmer, C., and Radespiel, R., "Reduced-Order Modeling of Vehicle Aerodynamics via Proper Orthogonal Decomposition," *SAE Int. J. Passeng. Cars - Mech. Syst.*, Vol. 12, 2019, pp. 225–236. https://doi.org/10.4271/06-12-03-0016.

[13] Shen, Y., Patel, H. C., Xu, Z., and Alonso, J. J., "Application of Multi-Fidelity Transfer Learning with Autoencoders for Efficient Construction of Surrogate Models," *AIAA SCITECH 2024 Forum*, 2024, pp. 1–19. https://doi.org/10.2514/6.2024-0013.

[14] Shen, Y. Huang, W. Wang, Z.-g. Xu, D.-f. and Liu, C.-Y. "A deep learning framework for aerodynamic pressure prediction on general three-dimensional configurations," *Physics of Fluids*, Vol. 35, No. 10, 2023, p. 107111. https://doi.org/10.1063/5.0172437, URL https://doi.org/10.1063/5.0172437.

[15] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. W., "Learning Mesh-Based Simulation with Graph Networks," , 2021. URL https://arxiv.org/abs/2010.03409.

[16] Catalani, G., Agarwal, S., Bertrand, X., Tost, F., Bauerheim, M., and Morlier, J., "Neural fields for rapid aircraft aerodynamics simulations," *Scientific Reports*, Vol. 14, No. 1, 2024, p. 25496. https://doi.org/10.1038/s41598-024-76983-w, URL https://doi.org/10.1038/s41598-024-76983-w.

[17] Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E., "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators," *Nature Machine Intelligence*, Vol. 3, No. 3, 2021, p. 218–229. https://doi.org/10.1038/s42256-021-00302-5, URL http://dx.doi.org/10.1038/s42256-021-00302-5.

[18] Ranade, R., Nabian, M. A., Tangsali, K., Kamenev, A., Hennigh, O., Cherukuri, R., and Choudhry, S., "DoMINO: A Decomposable Multi-scale Iterative Neural Operator for Modeling Large Scale Engineering Simulations," , 2025. URL https://arxiv.org/abs/2501.13350.

[19] Li, Z., Kovachki, N. B., Choy, C., Li, B., Kossaifi, J., Otta, S. P., Nabian, M. A., Stadler, M., Hundt, C., Azizzadenesheli, K., and Anandkumar, A., "Geometry-Informed Neural Operator for Large-Scale 3D PDEs," , 2023. URL https://arxiv.org/abs/2309.00583.

[20] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A., "Fourier Neural Operator for Parametric Partial Differential Equations," , 2021. URL https://arxiv.org/abs/2010.08895.

[21] Zhao, T., Qian, W., Lin, J., Chen, H., Ao, H., Chen, G., and He, L., "Learning Mappings from Iced Airfoils to Aerodynamic Coefficients Using a Deep Operator Network," *Journal of Aerospace Engineering*, Vol. 36, No. 5, 2023, p. 04023035. https://doi.org/10.1061/JAEEEZ.ASENG-4508, URL https://ascelibrary.org/doi/abs/10.1061/JAEEEZ.ASENG-4508.

[22] Alkin, B., Bleeker, M., Kurle, R., Kronlachner, T., Sonnleitner, R., Dorfer, M., and Brandstetter, J., "AB-UPT: Scaling Neural CFD Surrogates for High-Fidelity Automotive Aerodynamics Simulations via Anchored-Branched Universal Physics Transformers," , 2025. URL https://arxiv.org/abs/2502.09692.

[23] Shen, Y., and Alonso, J. J., "Performance Evaluation of a Graph Neural Network-Augmented Multi-Fidelity Workflow for Predicting Aerodynamic Coefficients on Delta Wings at Low Speed," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-2360, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-2360.

[24] Shen, Y., and Alonso, J., "Graph Neural Network-Guided Aerodynamic Shape Optimization for Conceptual Design of Supersonic Transport Wings," *AIAA AVIATION FORUM AND ASCEND 2025*, 2025. https://doi.org/10.2514/6.2025-3228, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-3228.

[25] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D., "Scaling Laws for Neural Language Models," , 2020. URL https://arxiv.org/abs/2001.08361.

[26] Liu, J., Mao, H., Chen, Z., Zhao, T., Shah, N., and Tang, J., "Towards Neural Scaling Laws on Graphs," , 2024. URL https://arxiv.org/abs/2402.02054.

[27] Bekemeyer, P., Hariharan, N., Wissink, A. M., and Cornelius, J., "Introduction of Applied Aerodynamics Surrogate Modeling Benchmark Cases," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-0036, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-0036.

[28] Seraj, S., and Martins, J. R. R. A., "Predicting the High-Angle-of-Attack Characteristics of a Delta Wing at Low Speed," *Journal of Aircraft*, Vol. 59, No. 4, 2022, pp. 1071–1081. https://doi.org/10.2514/1.C036618.

[29] Lukaczyk, T. W., Wendorff, A. D., Colonno, M., Economon, T. D., Alonso, J. J., Orra, T. H., and Ilario, C., "SUAVE: An Open-Source Environment for Multi-Fidelity Conceptual Vehicle Design," *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2015. https://doi.org/10.2514/6.2015-3087, URL https://arc.aiaa.org/doi/abs/10.2514/6.2015-3087.

[30] OpenVSP, "Open Vehicle Sketch Pad," https://openvsp.org/, 2025. Accessed: 2025-06-01.

[31] Anderson, J. D., *Fundamentals of aerodynamics, 6th Edition*, McGraw-Hill Education, 2017.

[32] Miranda, L. R., Elliott, R. D., and Baker, W. M., "A generalized vortex lattice method for subsonic and supersonic flow applications," *NASA. Langley Res. Center Vortex-Lattice Utilization*, 1977. URL https://ntrs.nasa.gov/citations/19780008059.

[33] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J., "SU2: An Open-Source Suite for Multiphysics Simulation and Design," *AIAA Journal*, Vol. 54, No. 3, 2016, pp. 828–846. https://doi.org/10.2514/1.J053813.

[34] Dacles-Mariani, J., Zilliac, G. G., Chow, J. S., and Bradshaw, P., "Numerical/experimental study of a wingtip vortex in the near field," *AIAA Journal*, Vol. 33, No. 9, 1995, pp. 1561–1568. https://doi.org/10.2514/3.12826, URL https://doi.org/10.2514/3.12826.

[35] Alauzet, F., and Spalart, P., "An Improved Smooth Rotation Correction for the Spalart–Allmaras Turbulence Model for Better Off-Body Vortex Prediction and Vortex–Vortex Interaction Effects," *International Journal of Computational Fluid Dynamics*, Vol. 38, No. 2-3, 2024, pp. 265–297. https://doi.org/10.1080/10618562.2024.2444914, URL https://doi.org/10.1080/10618562.2024.2444914.

[36] Widhalm, M., Stradtner, M., Schütte, A., Ghoreyshi, M., Jirásek, A., and Seidel, J., "Stability Derivatives Prediction Using Reduced-Order Models on Triple Delta Wing," *Journal of Aircraft*, Vol. 0, No. 0, 0, pp. 1–14. https://doi.org/10.2514/1.C038321, URL https://doi.org/10.2514/1.C038321.

[37] Di Fabbio, T., Rajkumar, K., Tangermann, E., and Klein, M., "Towards the understanding of vortex breakdown for improved RANS turbulence modeling," *Aerospace Science and Technology*, Vol. 146, 2024, p. 108973.

[38] Werner, M., Schütte, A., and Weiss, S., "Turbulence Model Effects on the Prediction of Transonic Vortex Interaction on a Multi-Swept Delta Wing," *AIAA SCITECH 2022 Forum*, 2022. https://doi.org/10.2514/6.2022-0295, URL https://arc.aiaa.org/doi/abs/10.2514/6.2022-0295.

[39] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y., "Graph Attention Networks," *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

[40] Wiersema, K., Ting, K.-Y., Nelson, C., and Livne, E., "Supersonic Configurations at Low Speeds(SCALOS): Correlation of CFD and Wind Tunnel Test Results," *AIAA SCITECH 2025 Forum*, 2025. https://doi.org/10.2514/6.2025-2362, URL https://arc.aiaa.org/doi/abs/10.2514/6.2025-2362.

[41] Grenville, A., and Guibas, L. J., "Spline Curves," , 1992. URL https://graphics.stanford.edu/courses/cs348a-08-winter/.

[42] Livne, E., Nelson, C. P., Soltani, R. M., Ting, K.-Y., Mavriplis, N., Wiersema, K., Cesnik, C. E. S., Kolmanovsky, I., Martins, J. R. R. A., Guimaraes, T. A. M., Cunis, T., Barbosa, G. C., Bertolin, R., Seraj, S., Monteiro, B. B., Alonso, J. J., Kroo, I., Shanbhag, T. K., Shen, Y., Magee, T. E., Hayes, P. J., Dorgan, A. J., and Khodadoust, A., "Compilation of Supersonic Configurations at Low Speeds (SCALOS) Final Report and Appendices A – F," Contractor or Grantee Report 20250006613, NASA - Langley Research Center, July 2025. URL https://ntrs.nasa.gov/citations/20250006613.

[43] Giunta, A. A., Balabanov, V., Haim, D., Grossman, B., Mason, W. H., Watson, L. T., and Haftka, R. T., "Multidisciplinary optimisation of a supersonic transport using design of experiments theory and response surface modelling," *The Aeronautical Journal*, Vol. 101, No. 1008, 1997, p. 347–356. https://doi.org/10.1017/S0001924000066045.

[44] Joe, S., and Kuo, F. Y., "Constructing Sobol Sequences with Better Two-Dimensional Projections," *SIAM Journal on Scientific Computing*, Vol. 30, No. 5, 2008, pp. 2635–2654. https://doi.org/10.1137/070709359, URL https://doi.org/10.1137/070709359.

[45] Herman, J., and Usher, W., "SALib: An open-source Python library for Sensitivity Analysis," *The Journal of Open Source Software*, Vol. 2, No. 9, 2017. https://doi.org/10.21105/joss.00097, URL https://doi.org/10.21105/joss.00097.

[46] Campolongo, F., Saltelli, A., and Cariboni, J., "From screening to quantitative sensitivity analysis. A unified approach," *Computer physics communications*, Vol. 182, No. 4, 2011, pp. 978–988.

[47] Saltelli, A., Tarantola, S., and Chan, K.-S., "A quantitative model-independent method for global sensitivity analysis of model output," *Technometrics*, Vol. 41, No. 1, 1999, pp. 39–56.

[48] Pointwise, "VSP2CFD," https://github.com/pointwise/VSP2CFD, 2021. Accessed Github Repository: 2025-09-30.

[49] Sutherland, W., "LII. The viscosity of gases and molecular force," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Vol. 36, No. 223, 1893, pp. 507–531. https://doi.org/10.1080/14786449308620508, URL https://doi.org/10.1080/14786449308620508.

[50] Jameson, A., Schmidt, W., and Turkel, E., "Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes," *14th Fluid and Plasma Dynamics Conference*, 1981. https://doi.org/10.2514/6.1981-1259.

[51] Liou, M.-S., and Steffen, C. J., "A New Flux Splitting Scheme," *Journal of Computational Physics*, Vol. 107, No. 1, 1993, pp. 23–39. https://doi.org/https://doi.org/10.1006/jcph.1993.1122, URL https://www.sciencedirect.com/science/article/pii/S0021999183711228.

[52] Hestness, J., Narang, S., Ardalani, N., Diamos, G., Jun, H., Kianinejad, H., Patwary, M. M. A., Yang, Y., and Zhou, Y., "Deep Learning Scaling is Predictable, Empirically," , 2017. URL https://arxiv.org/abs/1712.00409.

[53] Rosenfeld, J. S., Rosenfeld, A., Belinkov, Y., and Shavit, N., "A Constructive Prediction of the Generalization Error Across Scales," , 2019. URL https://arxiv.org/abs/1909.12673.

[54] Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L., "Training Compute-Optimal Large Language Models," , 2022. URL https://arxiv.org/abs/2203.15556.

[55] Li, C., Ye, Z., Pasini, M. L., Choi, J. Y., Wan, C., and Balaprakash, P., "Scaling Laws of Graph Neural Networks for Atomistic Materials Modeling," *Proceedings of the 62nd Annual ACM/IEEE Design Automation Conference*, IEEE Press, 2025. https://doi.org/10.1109/DAC63849.2025.11132864, URL https://doi.org/10.1109/DAC63849.2025.11132864.

[56] Sypetkowski, M., Wenkel, F., Poursafaei, F., Dickson, N., Suri, K., Fradkin, P., and Beaini, D., "On the Scalability of GNNs for Molecular Graphs," *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=klqhrq7fvB.

[57] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M., "Optuna: A Next-generation Hyperparameter Optimization Framework," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Association for Computing Machinery, New York, NY, USA, 2019, p. 2623–2631. https://doi.org/10.1145/3292500.3330701, URL https://doi.org/10.1145/3292500.3330701.

[58] Geiping, J., and Goldstein, T., "Cramming: Training a Language Model on a single GPU in one day." *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 202, edited by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, PMLR, 2023, pp. 11117–11143. URL https://proceedings.mlr.press/v202/geiping23a.html.

[59] Izsak, P., Berchansky, M., and Levy, O., "How to Train BERT with an Academic Budget," *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, edited by M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 10644–10652. https://doi.org/10.18653/v1/2021.emnlp-main.831, URL https://aclanthology.org/2021.emnlp-main.831/.