AMG: ADAPTIVE MULTI-POINT GUIDANCE FOR AERODYNAMIC DESIGN VIA DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Aerodynamic inverse design aims to find a wing geometry that achieves specified lift and drag coefficients across multiple flight conditions. Diffusion models have recently emerged as a promising tool for this task, but existing approaches coordinate multiple operating conditions through manually tuned or heuristic weighting schemes, which cannot adaptively resolve conflicts among targets. We propose Adaptive Multi-point Guidance (AMG), an inference-time guidance framework that steers a pretrained diffusion model by solving a linear system in the latent space. At each denoising step, AMG evaluates the denoised prediction with a differentiable surrogate model, constructs the physical Jacobian of the aerodynamic coefficients with respect to the latent variable, and solves for a latent displacement whose pseudo-inverse solution automatically determines the contribution of each target. This formulation replaces the noise-space scalar weighting of classifier guidance with a latent-space vector displacement, and enables adaptive coordination of competing aerodynamic objectives without manual weight tuning. On a set of multi-point design tasks constructed from the SuperWing dataset, AMG reduces the average MAPE by 69% relative to the best-performing baseline among those considered in this work, where the MAPE is evaluated by the same surrogate model used for guidance and reported at the best intermediate step of sampling. These results suggest that AMG has the potential to be applied in practical aerodynamic design.

1 INTRODUCTION

Aerodynamic inverse design is a cornerstone of aircraft wing design, where the goal is to find a geometry that achieves desired aerodynamic performance, typically specified by lift coefficient C_L and drag coefficient C_D. In practice, a wing must perform well across multiple flight conditions, such as different angles of attack (AoA) and Mach numbers. This multi-point inverse design problem is challenging because the targets at different conditions often conflict: a geometry optimized for one condition may degrade performance at another.

Traditional approaches to aerodynamic shape optimization fall into two broad categories: gradient-based methods and surrogate-assisted evolutionary algorithms. Gradient-based methods compute the sensitivity of an objective function with respect to design variables and iteratively update the geometry. Among them, direct differentiation methods solve a linearized flow problem for each design variable, making the gradient computation cost proportional to the number of design variables (Papadimitriou & Giannakoglou, 2008). The adjoint method, pioneered by Jameson (1988), avoids this cost by solving a single adjoint equation that is independent of the number of design variables, and has since become the standard approach for high-fidelity aerodynamic shape optimization (Peter & Dwight, 2010). On the other hand, surrogate-assisted evolutionary algorithms replace expensive computational fluid dynamics (CFD) evaluations with cheap surrogate models, such as Kriging or neural networks, to enable global search at reduced computational cost (Jin, 2011; Han et al., 2020).

Recent advances in generative modeling, particularly diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020), have opened a fundamentally different avenue for design synthesis. By learning the distribution of valid geometries from data, diffusion models can generate diverse and high-quality designs without requiring an initial guess. Classifier guidance (CG) (Dhariwal & Nichol, 2021) uses gradients from a pretrained classifier to steer the sampling process, while classifier-free guidance

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(CFG) (Ho & Salimans, 2022) jointly trains conditional and unconditional models, eliminating the need for an external classifier. In the context of aerodynamic design, CcDPM (Zhao et al., 2024) was the first to introduce multi-point inverse design into the aerodynamic domain, and Airfoil-DDPM (Wen et al., 2025) subsequently proposed a distance-adaptive sampling strategy for multi-condition airfoil design. Beyond diffusion models, Dflow-SUR (Yang et al., 2025) applies flow matching to aerodynamic inverse design by optimizing the initial noise through the entire generation trajectory.

Despite these advances, applying generative models to multi-point aerodynamic inverse design remains challenging. Existing approaches combine guidance signals through manually tuned or fixed weighting schemes, leaving the relative contribution of each operating condition to be determined by hand. When targets across different flight conditions impose competing requirements on the same geometry, such heuristics offer no principled way to determine how much each target should contribute to the update, and the chosen weights may over-emphasize some conditions at the expense of others. These limitations motivate a guidance framework that can automatically balance competing objectives without manual tuning, while steering a diffusion model toward multiple aerodynamic targets within a single sampling process.

In this paper, we propose Adaptive Multi-point Guidance (AMG), an inference-time guidance framework for multi-condition aerodynamic inverse design. Given a diffusion model, a differentiable surrogate model, and a free-form deformation (FFD) parameterization of the wing geometry, AMG steers the sampling trajectory at inference time to satisfy multiple C_L and C_D targets simultaneously. The key idea is to reconstruct the local Jacobian of the aerodynamic coefficients with respect to the latent variables at each denoising step, and solve a linear system that determines how much each target should contribute to the latent update. This mechanism automatically balances competing targets without manual weighting. We evaluate AMG on a set of multi-point design tasks with different target combinations and operating conditions. Experimental results show that AMG generates geometries that satisfy multiple C_L and C_D targets, and consistently outperforms the baselines considered in our experiments.

Our contributions are summarized as follows:

- We propose AMG, an inference-time guidance framework for multi-point aerodynamic inverse design that steers a diffusion model toward multiple targets within a single sampling process.
- We introduce a Jacobian reconstruction and linear displacement solve that adaptively balances competing aerodynamic targets without manual weighting.
- We demonstrate that AMG generates 3D wing geometries satisfying multiple targets across different operating conditions, and outperforms the compared baselines on a set of multi-point design tasks.

2 Related Work

2.1 Diffusion Models and Guidance Methods

Diffusion models were introduced by Sohl-Dickstein et al. (2015) and later made practical by Ho et al. (2020) through a noise-prediction reparameterization of the reverse process, and have since become the dominant generative paradigm across image, audio, and 3D shape synthesis. To steer a diffusion model toward a desired output, Dhariwal & Nichol (2021) introduced CG and Ho & Salimans (2022) proposed CFG, which respectively use the gradient of an external classifier and the difference between conditional and unconditional predictions to modify the sampling trajectory. CG and CFG were originally designed for a single conditioning signal. Follow-up works such as TFG (Ye et al., 2024) and independent condition guidance (Sadat et al., 2025) remain single-objective. Universal Guidance (Bansal et al., 2024) supports multiple differentiable objectives, but the objectives are combined into a single loss with user-defined priorities. Split-CFG (Karras et al., 2024) allows independent control over multiple conditioning signals, but the per-condition weights are manually set and the result depends on the ordering of conditions. AtteConDA (Noguchi, 2026) selects the most effective local condition at each location via a learned attention module, rather than coordinating all conditions simultaneously. These methods are not directly applicable to multi-

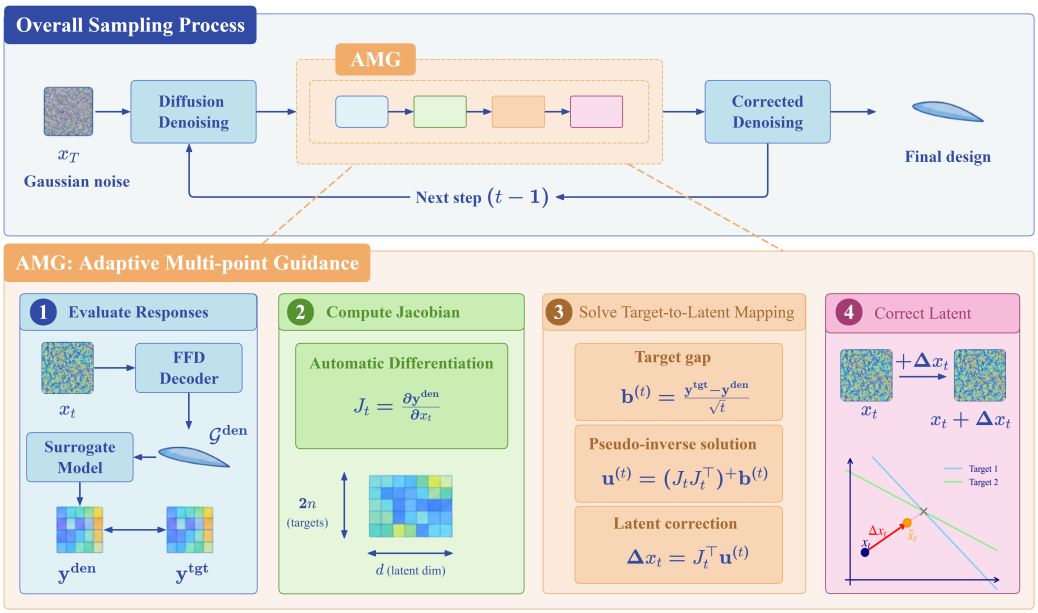


Figure 1: Overview of AMG. The diffusion trajectory produces a denoised latent μ_t at step t, which is decoded by the FFD decoder and evaluated by a differentiable surrogate model under all operating conditions, yielding $\mathbf{y}^{\text{den}}$. AMG constructs the desired target change $\mathbf{b}^{(t)}$, computes the Jacobian J_t via automatic differentiation, solves for the adaptive weights $\mathbf{u}^{(t)}$, and applies the resulting latent displacement Δx_t to obtain the corrected latent $\tilde{x}_t = x_t + \Delta x_t$, which is passed through a corrected denoising step. This process repeats for $t = T, \ldots, 1$, and the final design is decoded from x_0.

point aerodynamic inverse design, where a single geometry must satisfy multiple continuous-valued physical constraints that may conflict, which is the problem our AMG method addresses.

2.2 Generative Models for Aerodynamic Design

In aerodynamic design, early generative models were based on conditional GANs (CGANs), whose conditioning mechanism is designed for discrete categorical labels. Continuous conditional GANs (CcGANs) (Ding et al., 2021) introduced Vicinal Risk Minimization to handle continuous conditions, but remain limited to single-point design. Diffusion models have since become the dominant paradigm: CcDPM (Zhao et al., 2024) was the first to bring diffusion models into engineering inverse design, proposing a continuous conditional diffusion model with a multi-point sampling framework, yet its multi-point guidance relies on fixed user-defined weights. Airfoil-DDPM (Wen et al., 2025) extends this with a distance-adaptive multi-objective sampling scheme that dynamically allocates weights, but targets that deviate significantly from the remaining conditions receive negligible weight and are therefore not actively satisfied. Beyond diffusion, flow matching has also been explored, e.g., Dflow-SUR (Yang et al., 2025) performs gradient-driven optimization in the initial source space, but it cannot adaptively adjust target contributions during sampling. Overall, existing methods fail to adequately resolve conflicts among multiple targets, and their success heavily depends on manual tuning of guidance weights, often requiring multiple trials. Our AMG method provides a way to address these limitations by leveraging physical gradients to adaptively determine the contribution of each target, eliminating manual weight tuning.

3 Method

The overall pipeline is illustrated in Figure 1. Starting from Gaussian noise x_T, AMG iterates over denoising steps $t = T, \ldots, 1$. At each step, it evaluates the responses of the denoised geometries, computes the Jacobian via automatic differentiation, solves for the adaptive weights and the latent displacement, and applies the corrected denoising step to produce x_{t-1}. We first define the problem

and the components, then present each of these steps in detail, and finally summarize the complete procedure.

3.1 Problem Setup

We consider the problem of multi-point aerodynamic inverse design, where a single geometry must satisfy specified aerodynamic targets across multiple operating conditions. Given n operating conditions $\{(\mathrm{AoA}_m, \mathrm{Ma}_m)\}_{m=1}^n$ and target coefficients $C_{L,m}^{\mathrm{tgt}}, C_{D,m}^{\mathrm{tgt}}$, the goal is to generate a single 3D wing geometry $\mathcal{G}$ that simultaneously satisfies

$$C_L(\mathcal{G}, \mathrm{AoA}_m, \mathrm{Ma}_m) = C_{L,m}^{\mathrm{tgt}}, \quad C_D(\mathcal{G}, \mathrm{AoA}_m, \mathrm{Ma}_m) = C_{D,m}^{\mathrm{tgt}}, \quad \forall m \in \{1, \dots, n\}, \tag{1}$$

where C_L and C_D denote the lift and drag coefficients of geometry $\mathcal{G}$ evaluated at the given operating condition by a surrogate model. We concatenate the targets into a single vector

$$\mathbf{y}^{\mathrm{tgt}} = \left[C_{L,1}^{\mathrm{tgt}}, C_{D,1}^{\mathrm{tgt}}, \dots, C_{L,n}^{\mathrm{tgt}}, C_{D,n}^{\mathrm{tgt}}\right]^\top \in \mathbb{R}^{2n}. \tag{2}$$

We use three components:

- a diffusion model ϵ_θ that performs one denoising step, denoted by $\mu_t = \mathrm{denoise}(x_t; \epsilon_\theta, t)$;
- an FFD decoder $\mathcal{D}$ that maps a latent variable to a geometry, $\mathcal{G} = \mathcal{D}(x)$;
- a differentiable surrogate model $\mathcal{S}$ that predicts aerodynamic coefficients, $(\hat{C}_L, \hat{C}_D) = \mathcal{S}(\mathcal{G}, \mathrm{AoA}, \mathrm{Ma})$.

At each step t, we evaluate the denoised prediction:

$$\mathbf{y}^{\mathrm{den}} = \left[\mathcal{S}(\mathcal{D}(\mu_t), \mathrm{AoA}_m, \mathrm{Ma}_m)\right]_{m=1}^n \in \mathbb{R}^{2n}. \tag{3}$$

The Jacobian of the denoised prediction with respect to x_t is

$$J_t = \frac{\partial \mathbf{y}^{\mathrm{den}}}{\partial x_t} \in \mathbb{R}^{2n \times d}. \tag{4}$$

3.2 Components

3.2.1 Classifier Guidance Diffusion Model

Our method builds on a variant of classifier guidance (CG) (Dhariwal & Nichol, 2021). Let x_t be the latent at step t, with t decreasing from T to 1. The noise schedule is $\{\beta_t\}$ with $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^t \alpha_s$.

In standard DDPM (Ho et al., 2020), the reverse step is stochastic:

$$x_{t-1} = \mu_\theta(x_t, t) + \sigma_t z, \quad z \sim \mathcal{N}(0, \mathbf{I}), \tag{5}$$

where $\mu_\theta(x_t, t)$ is the posterior mean predicted by the network and σ_t the posterior standard deviation. In this work, we use the deterministic DDPM posterior-mean sampler, obtained by setting $z = 0$, so that

$$x_{t-1} = \mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right). \tag{6}$$

For brevity, we denote this deterministic posterior mean by $\mu_t = \mathrm{denoise}(x_t; \epsilon_\theta, t) = \mu_\theta(x_t, t)$.

In standard CG, a classifier $p_\phi(y \mid x_t)$ guides the noise prediction:

$$\hat{\epsilon}_\theta(x_t, t) = \epsilon_\theta(x_t, t) - \lambda \sqrt{1 - \bar{\alpha}_t}\, \nabla_{x_t} \log p_\phi(y \mid x_t), \tag{7}$$

where λ is a user-specified guidance weight. The next latent is

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \hat{\epsilon}_\theta(x_t, t) \right). \tag{8}$$

For multiple targets, a standard extension sums classifier gradients with weights λ_k. In AMG, we do not use this guidance path. Instead, we guide the sampling process by solving a linear system in the latent space, as described in Section 3.3.

3.2.2 FREE-FORM DEFORMATION

We represent the wing geometry using FFD (Sederberg & Parry, 1986), which embeds a base shape $\mathcal{G}_{\text{base}}$ in a lattice of $N_x \times N_y \times N_z$ control points. Each control point (a, b, c) has a displacement $\boldsymbol{\delta}_{abc} \in \mathbb{R}^3$, and the deformed position of a point $\mathbf{s}$ on the base shape is obtained by B-spline interpolation with weights $W_{abc}(\mathbf{s})$ satisfying $\sum_{a,b,c} W_{abc}(\mathbf{s}) = 1$.

The FFD decoder $\mathcal{D}$ maps a latent variable $x \in \mathbb{R}^d$ to a geometry:

$$\mathcal{G} = \mathcal{D}(x) = \left\{ \mathbf{s} + \sum_{a,b,c} W_{abc}(\mathbf{s})\, \boldsymbol{\delta}_{abc}(x) \;\middle|\; \mathbf{s} \in \mathcal{G}_{\text{base}} \right\}, \tag{9}$$

i.e., $\mathcal{G}$ is the set of all deformed points. Here $\boldsymbol{\delta}_{abc}(x)$ is obtained by reshaping x into an $N_x \times N_y \times N_z \times 3$ tensor and applying the normalization used during training. Its input is the latent variable x, and its output is the geometry $\mathcal{G}$. The decoder is differentiable end-to-end, so gradients can be propagated from any function of $\mathcal{G}$ back to x.

3.2.3 DIFFERENTIABLE SURROGATE MODEL

We use a neural surrogate model $\mathcal{S}$ trained on CFD data. Its input is a geometry $\mathcal{G}$ and an operating condition $(\mathrm{AoA}, \mathrm{Ma})$, and its output is the corresponding lift and drag coefficients:

$$(\hat{C}_L, \hat{C}_D) = \mathcal{S}(\mathcal{G}, \mathrm{AoA}, \mathrm{Ma}). \tag{10}$$

3.3 ADAPTIVE MULTI-POINT GUIDANCE

Standard CG adds a classifier gradient to the noise prediction with a manually tuned weight. With multiple targets, this is problematic: competing requirements can suppress one another, and a fixed weight cannot adapt as sampling evolves. AMG instead treats the gradient as local sensitivity information: within one denoising step, the response of each coefficient to a small latent displacement is approximated linearly. Specifying how much each target should change then determines the displacement through a linear system, which yields both each target's contribution and the displacement itself, turning multi-target coordination into solving a linear system.

Linearity assumption. Let $x_t \in \mathbb{R}^d$ denote the latent variable at denoising step t, where d is the dimension of the latent space. Recall that the denoised prediction $\mathbf{y}^{\text{den}}$ is obtained by first applying the one-step denoising operator to obtain $\mu_t = \mathrm{denoise}(x_t; \epsilon_\theta, t)$, then decoding it into a geometry via the FFD decoder $\mathcal{D}$, and finally evaluating the aerodynamic coefficients via the surrogate model $\mathcal{S}$ under all operating conditions. Recall that the Jacobian J_t defined in the problem setup has rows $J_{t,k} = \partial y_k^{\text{den}} / \partial x_t$, each specifying how the k-th aerodynamic coefficient of the denoised prediction responds to an infinitesimal change in the latent variable.

By first-order Taylor expansion,

$$\mathbf{y}^{\text{den}}(x_t + \Delta x_t) \approx \mathbf{y}^{\text{den}}(x_t) + J_t\, \Delta x_t. \tag{11}$$

We assume that this first-order approximation remains accurate for the finite displacement computed at each denoising step. Under this assumption, specifying the desired change in each aerodynamic coefficient determines Δx_t through a linear system.

Gradient computation and guidance path. AMG aims to make the geometry produced by the next denoising step satisfy the target aerodynamic coefficients. This goal determines where the guidance signal must be evaluated and how it must be applied.

In standard CG, the goal is to sample from the conditional distribution $p(x_t \mid y)$. By Bayes' rule, the conditional score decomposes into the unconditional score plus the classifier gradient, and since the score is equivalent to the noise prediction, the guidance is naturally applied to the noise prediction.

AMG has a different goal: we want the denoised latent to move toward the target. We therefore compute the guidance signal on the denoised prediction $\mathbf{y}^{\text{den}}$. Concretely, we first apply the one-step denoising operator to obtain μ_t, decode it into a geometry via the FFD decoder, and evaluate the aerodynamic coefficients to obtain $\mathbf{y}^{\text{den}}$. The Jacobian J_t is then computed by backpropagating

through the surrogate model, the FFD decoder, and the denoising step, so that $J_{t,k}$ describes how the k-th coefficient of the denoised prediction responds to a change in x_t.

Because the gradient is taken with respect to x_t, the resulting correction is naturally a displacement of the latent variable:

$$\tilde{x}_t = x_t + \Delta x_t. \tag{12}$$

This is the essential difference from standard CG, which modifies the noise prediction. Applying the correction in the latent space allows the displacement to be determined by solving a linear system, which enables adaptive coordination across multiple targets.

Desired target change. Let y_k^{den} and y_k^{tgt} denote the k-th components of $\mathbf{y}^{\text{den}}$ and $\mathbf{y}^{\text{tgt}}$, respectively. Under the linearity assumption, the displacement Δx_t is determined by how much we want the denoised prediction to change in this step. We therefore need to specify a desired change $b_k^{(t)}$ for each target $k \in \{1, \ldots, 2n\}$.

The most direct choice is to close the entire gap between the target and the denoised prediction in a single step, i.e., $b_k^{(t)} = y_k^{\text{tgt}} - y_k^{\text{den}}$. This would produce a large displacement and violate the linearity assumption that the system relies on. A natural way to avoid this is to spread the gap uniformly over the remaining steps, i.e., $b_k^{(t)} = (y_k^{\text{tgt}} - y_k^{\text{den}})/t$. However, this has a mathematical consequence: because the gap is only fully closed when $t = 1$, the target is approached only near the end of the trajectory, and the guidance has little opportunity to correct for inaccuracies that arise along the way. We therefore adopt an intermediate scaling with exponent $1/2$,

$$b_k^{(t)} = \frac{y_k^{\text{tgt}} - y_k^{\text{den}}}{\sqrt{t}}. \tag{13}$$

Adaptive weight solve. We now have the Jacobian J_t and the desired change

$$\mathbf{b}^{(t)} = [b_1^{(t)}, \ldots, b_{2n}^{(t)}]^\top \in \mathbb{R}^{2n}. \tag{14}$$

Under the linearity assumption, a displacement Δx_t induces a change $J_t \Delta x_t$ in the denoised prediction. We want this change to match $\mathbf{b}^{(t)}$, which gives the linear system

$$J_t \Delta x_t \approx \mathbf{b}^{(t)}. \tag{15}$$

We solve this system using the Moore–Penrose pseudo-inverse of J_t, denoted by J_t^+:

$$\Delta x_t = J_t^+ \mathbf{b}^{(t)} = J_t^\top (J_t J_t^\top)^+ \mathbf{b}^{(t)}. \tag{16}$$

This expression can be written as

$$\Delta x_t = J_t^\top \mathbf{u}^{(t)} = \sum_{k=1}^{2n} u_k^{(t)} J_{t,k}^\top, \quad \mathbf{u}^{(t)} = (J_t J_t^\top)^+ \mathbf{b}^{(t)}, \tag{17}$$

where $\mathbf{u}^{(t)} \in \mathbb{R}^{2n}$ is the adaptive weight vector and $J_{t,k}^\top$ is the transpose of the k-th row of J_t. Its k-th entry $u_k^{(t)}$ determines the coefficient assigned to the k-th target when the displacement is expressed as a linear combination of the transposed rows of J_t.

3.4 Overall Algorithm

Algorithm 1 summarizes the complete AMG procedure. Starting from Gaussian noise x_T, it iterates over denoising steps $t = T, \ldots, 1$. At each step, it computes the one-step denoised mean μ_t, decodes the geometry $\mathcal{G}^{\text{den}}$, and evaluates it with the surrogate model under all operating conditions to obtain $\mathbf{y}^{\text{den}}$. The Jacobian $J_t = \partial \mathbf{y}^{\text{den}} / \partial x_t$ is computed by automatic differentiation through the surrogate, the FFD decoder, and the denoising step. The desired target change $\mathbf{b}^{(t)}$ defines the linear system $J_t \Delta x_t \approx \mathbf{b}^{(t)}$, which is solved via the pseudo-inverse to obtain the adaptive weights $\mathbf{u}^{(t)}$ and the displacement $\Delta x_t = J_t^\top \mathbf{u}^{(t)}$. The displacement is applied to obtain $\tilde{x}_t = x_t + \Delta x_t$, and a corrected denoising step produces x_{t-1}.

The algorithm tracks the best evaluation error along the trajectory. Since the diffusion model tends to regress toward the data distribution in the late stage of sampling, the best error is not necessarily achieved at the final step, so we report the best-step error instead of the final-step error.

Algorithm 1 Adaptive Multi-point Guidance (AMG)

Require: Diffusion model ϵ_θ, surrogate $\mathcal{S}$, FFD decoder $\mathcal{D}$, conditions $\{(\mathrm{AoA}_m, \mathrm{Ma}_m)\}_{m=1}^n$, targets $\mathbf{y}^{\text{tgt}} \in \mathbb{R}^{2n}$
Ensure: Final geometry $\mathcal{G}_{\text{final}}$, best geometry $\mathcal{G}^\star$, best step $t^\star$
Sample $x_T \sim \mathcal{N}(0, \mathbf{I})$
Initialize best_error $\leftarrow \infty$, $t^\star \leftarrow T$, $\mathcal{G}^\star \leftarrow \mathcal{D}(x_T)$
for $t = T, \ldots, 1$ **do**
  $\mu_t \leftarrow \mathrm{denoise}(x_t; \epsilon_\theta, t)$
  $\mathcal{G}^{\text{den}} \leftarrow \mathcal{D}(\mu_t)$
  $\mathbf{y}^{\text{den}} \leftarrow [\mathcal{S}(\mathcal{G}^{\text{den}}, \mathrm{AoA}_m, \mathrm{Ma}_m)]_{m=1}^n$
  Compute $J_t = \partial \mathbf{y}^{\text{den}} / \partial x_t$
  Construct $\mathbf{b}^{(t)} \in \mathbb{R}^{2n}$ with components $b_k^{(t)} \leftarrow \dfrac{y_k^{\text{tgt}} - y_k^{\text{den}}}{\sqrt{t}}$, for $k = 1, \ldots, 2n$
  $\mathbf{u}^{(t)} \leftarrow (J_t J_t^\top)^+ \mathbf{b}^{(t)}$
  $\Delta x_t \leftarrow J_t^\top \mathbf{u}^{(t)}$
  $\tilde{x}_t \leftarrow x_t + \Delta x_t$
  $x_{t-1} \leftarrow \mathrm{denoise}(\tilde{x}_t; \epsilon_\theta, t)$
  Evaluate the decoded geometry $\mathcal{D}(x_{t-1})$ with the surrogate $\mathcal{S}$ and compute the evaluation error
  if evaluation error $<$ best_error **then**
    best_error $\leftarrow$ evaluation error
    $t^\star \leftarrow t - 1$
    $\mathcal{G}^\star \leftarrow \mathcal{D}(x_{t-1})$
  end if
end for
$\mathcal{G}_{\text{final}} \leftarrow \mathcal{D}(x_0)$
return $\mathcal{G}_{\text{final}}, \mathcal{G}^\star, t^\star$

4 EXPERIMENTS

To validate the effectiveness of our method, we conduct experiments on a 3D wing dataset and compare against five baselines. Our experiments aim to answer the following questions.

Research Questions

RQ1: Compared with existing baselines, does AMG better satisfy multi-point aerodynamic design targets? (answered in Section 4.6)
RQ2: Does adaptive weighting outperform uniform weights across targets? (answered in Section 4.7)
RQ3: How does the scaling exponent in the demand vector affect the final design accuracy? (answered in Section 4.8)
RQ4: Do the generated geometries remain physically valid wing shapes across different tasks? (examined in Appendix H)

4.1 DATASET

We conduct experiments on the SuperWing dataset (Yang et al., 2026), a collection of 3D wing geometries together with their surface flow fields computed over a range of operating conditions. Each entry provides the surface point cloud of a wing geometry and the corresponding surface pressure and skin-friction coefficients, from which the lift coefficient C_L and drag coefficient C_D at each operating condition can be computed.

The SuperWing dataset serves three purposes in our experiments: it is used to train the surrogate model, to provide FFD-parameterized shapes for training the diffusion model, and to generate the multi-point design tasks used for evaluation. Details are provided in Appendix A.

4.2 FFD Parameterization

We parameterize each wing geometry using FFD with a control lattice of size $9 \times 2 \times 11$ in the streamwise, normal, and spanwise directions, respectively, and B-spline degrees $(3, 1, 3)$ along the three directions. This yields $9 \times 2 \times 11 \times 3 = 594$ degrees of freedom, which form the latent space of the diffusion model. Details are provided in Appendix B.

4.3 Surrogate Model

All aerodynamic evaluations, both during training and at inference, are performed by a Transolver surrogate model (Wu et al., 2024) trained on the SuperWing dataset. The surrogate takes a wing geometry, represented by its surface center points, and an operating condition $(\mathrm{AoA}, \mathrm{Ma})$ as input, and outputs the corresponding pressure and friction fields, which are integrated to obtain C_L and C_D. The surrogate is fully differentiable, which allows gradients to be propagated from the aerodynamic coefficients back to the geometry and then to the latent variable. The same surrogate is used for every baseline and for AMG, so that all methods are evaluated under the same aerodynamic model. Whether the resulting ranking transfers to higher-fidelity CFD is a separate question that depends on the surrogate-to-CFD gap, which is not studied here. Details are provided in Appendix C.

4.4 Baselines

We compare AMG with five baselines: CcDPM (Zhao et al., 2024), Airfoil-DDPM (Wen et al., 2025), standard CG (Dhariwal & Nichol, 2021), NRCG (Vaeth et al., 2025), and Dflow-SUR (Yang et al., 2025). All baselines use the same diffusion backbone, FFD decoder, and surrogate model as AMG. The only differences are the training procedure or the guidance strategy. Some related generative methods are not included as baselines because their mechanisms are designed for different settings or are complementary to AMG rather than alternatives to it. Details are provided in Appendix D.

4.5 Experimental Protocol and Metric

To construct multi-point design tasks, we partition the angle-of-attack and Mach number ranges into 5 intervals each, yielding 25 bins. For each bin, we compute the mean and standard deviation of the corresponding C_L and C_D values. A design point is generated by sampling $(\mathrm{AoA}, \mathrm{Ma})$ uniformly from the dataset range and then sampling C_L and C_D from within one standard deviation of the bin mean. Each task consists of two such design points that must be satisfied by a single geometry. We generate 16 tasks and use the same set for all methods (Appendix F).

For each task, we compute the MAPE across all targets:

$$\text{MAPE} = \frac{1}{2n} \sum_{k=1}^{2n} \frac{|y_k - y_k^{\text{tgt}}|}{|y_k^{\text{tgt}}|} \times 100\%, \tag{18}$$

where y_k is the surrogate-predicted coefficient and y_k^{tgt} the target. No target is zero, so the denominator is always defined. The best step is the one minimizing this MAPE during sampling, and we average the best-step MAPE over all 16 tasks. We also count geometries that are severely distorted by visual inspection; all generated geometries and the invalid cases are shown in Appendix H.

4.6 Main Results

Table 1 reports the best-step MAPE and the number of visually invalid geometries for all methods on the 16 multi-point tasks. AMG achieves an average best-step MAPE of 4.32%, approximately 69% lower than the best baseline (Standard CG, 14.03%). The invalid count shows that this accuracy is not obtained at a significant cost in geometric validity. A comparison between best-step and final-step MAPE is provided in Appendix G.

Table 1: Average MAPE (%) over 16 multi-point design tasks. Lower is better.

Method	MAPE (%)	Invalid / 16
CcDPM	27.61	0
Airfoil-DDPM	26.81	0
Standard CG	14.03	1
NRCG	22.76	1
Dflow-SUR	18.63	8
AMG (ours)	4.32	1

4.7 Adaptive Weighting

We compare AMG with a fixed-weight variant that uses the squared-error loss to compute the gradient, but applies a fixed guidance weight instead of solving for the adaptive weight vector. Table 2 confirms that adaptive weighting is essential. We do not separately ablate the guidance path, because the path and the solver are coupled: AMG solves in the latent space while standard CG corrects the noise prediction, and their effective step sizes differ by a factor that depends on t. The comparison with standard CG in Table 1 thus serves as the reference for both paths under their own designs.

Table 2: Ablation on adaptive weighting. Average MAPE (%) over 16 tasks.

Variant	MAPE (%)	Invalid / 16
Fixed weights	14.21	1
AMG	4.32	1

4.8 Scaling Schedule

We compare the scaling exponent $1/2$ in the demand vector with two alternatives: no scaling ($b_k^{(t)} = y_k^{\text{tgt}} - y_k^{\text{den}}$) and uniform scaling ($b_k^{(t)} = (y_k^{\text{tgt}} - y_k^{\text{den}})/t$). Table 3 reports the results.

Table 3: Ablation on the scaling exponent in the demand vector. Average MAPE (%) over 16 tasks.

Scaling	MAPE (%)	Invalid / 16
No scaling	2.34	6
$1/t$	10.95	1
$1/\sqrt{t}$ (ours)	4.32	1

Without scaling, the per-step displacement is not bounded. Although this leads to a lower average MAPE, the number of invalid geometries is much higher than for the other two variants, as listed in Table 3 and illustrated in Appendix H. With uniform scaling $1/t$, the per-step correction decays too quickly and the target is approached only near the end of sampling, resulting in the highest error among the three variants. The exponent $1/2$ provides a balance between target accuracy and geometric regularity, and is therefore adopted in AMG.

5 Discussion

AMG is a guidance strategy rather than a generative model, and it can be combined with other guidance-based generative methods to coordinate multiple design targets. Its practical performance depends on the accuracy of the surrogate model that provides the guidance signal, but the surrogate itself is not the subject of this work.

Two limitations of the method remain. First, the method does not explicitly constrain the magnitude of the latent displacement, so a large update may produce a geometry that is smooth but not physically valid. Second, the linearity assumption relies on the FFD parameterization, which provides a reliable geometry even in early denoising steps; whether the same holds for other parameterizations is not verified. Future work will address these issues by regularizing the displacement and testing alternative shape representations.

6 Conclusion

We presented AMG, an inference-time guidance framework for multi-point aerodynamic inverse design with diffusion models. AMG steers sampling by solving a linear system in the latent space: at each denoising step, it evaluates the denoised prediction with a differentiable surrogate model, constructs the physical Jacobian of the aerodynamic coefficients, and computes a latent displacement whose pseudo-inverse solution adaptively determines the contribution of each target. This replaces the noise-space scalar weighting of standard classifier guidance with a latent-space vector displacement, eliminating manual weight tuning. Experiments on multi-point design tasks constructed from a 3D wing dataset show that AMG achieves a lower average MAPE than the baselines considered in this work. These results suggest that AMG has the potential to be applied in practical aerodynamic design.

AI USE STATEMENT

In this work, we used generative AI tools for code generation, writing assistance, and literature survey. We have not used generative AI tools for generating research ideas, methods, experimental data, or results, and the remaining required disclosure tasks are not applicable to this work. The tasks listed above fall under recommended disclosure. We have reviewed all AI-assisted work: code was tested and verified for correctness, literature references were checked manually, and all AI-assisted text was reviewed and edited by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have made efforts to ensure the reproducibility of this work. The supplementary materials include the source code for AMG and for all baselines implemented in this paper. The code covers the diffusion model, the FFD parameterization, the surrogate model, the AMG sampling procedure, and the evaluation pipeline. The datasets used in our experiments are publicly available, and the data processing steps are described in Appendix A. The FFD configuration, surrogate model details, diffusion backbone, and baseline implementations are provided in Appendices B, C, and D. The exact list of multi-point design tasks used in our experiments is given in Appendix F.

REFERENCES

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Roni Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 51304–51323, 2024.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Xin Ding, Yongwei Wang, Zuheng Xu, William J Welch, and Z Jane Wang. Ccgan: Continuous conditional generative adversarial networks for image generation. In *International conference on learning representations*, 2021.

ZH Han, CZ Xu, JL Qiao, Fei Liu, JB Chi, GY Meng, KS Zhang, and WP Song. Recent progress of efficient global aerodynamic shape optimization using surrogate-based approach. *Acta Aeronautica et Astronautica Sinica*, 41(5):623344, 2020.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Antony Jameson. Aerodynamic design via control theory. *Journal of scientific computing*, 3(3): 233–260, 1988.

Yaochu Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

Johanna Karras, Yingwei Li, Nan Liu, Luyang Zhu, Innfarn Yoo, Andreas Lugmayr, Chris Lee, and Ira Kemelmacher-Shlizerman. Fashion-vdm: Video diffusion model for virtual try-on. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024.

Charles A Mader, Gaetan KW Kenway, Anil Yildirim, and Joaquim RRA Martins. Adflow: An open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization. *Journal of Aerospace Information Systems*, 17(9):508–527, 2020.

Shogo Noguchi. Atteconda: Attention-based conflict suppression in multi-condition diffusion models and synthetic data augmentation. *arXiv preprint arXiv:2605.09425*, 2026.

Dimitrios I Papadimitriou and Kyriakos C Giannakoglou. Aerodynamic shape optimization using first and second order adjoint and direct approaches. *Archives of Computational Methods in Engineering*, 15(4):447–488, 2008.

Jacques EV Peter and Richard P Dwight. Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches. *Computers & Fluids*, 39(3):373–391, 2010.

Seyedmorteza Sadat, Manuel Kansy, Otmar Hilliges, and Romann Weber. No training, no problem: Rethinking classifier-free guidance for diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 76833–76858, 2025.

Thomas W Sederberg and Scott R Parry. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pp. 151–160, 1986.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Philipp Vaeth, Dibyanshu Kumar, Benjamin Paassen, and Magda Gregorová. Diffusion classifier guidance for non-robust classifiers. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 206–221. Springer, 2025.

Zhe Wen, Yining Lian, Lifang Zeng, Chengwen Tian, Jun Li, and Xueming Shao. Airfoil-ddpm: A flexible airfoil generative design method using a multi-objective sampling based diffusion model. *Aerospace Science and Technology*, pp. 111070, 2025.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024.

Aobo Yang, Zhen Wei, Rhea Liem, and Pascal Fua. Dflow-sur: Enhancing generative aerodynamic inverse design using differentiation throughout flow matching. *arXiv preprint arXiv:2512.08336*, 2025.

Yunjia Yang, Weishao Tang, Mengxin Liu, Nils Thuerey, Yufei Zhang, and Haixin Chen. Superwing: a comprehensive transonic wing dataset for data-driven aerodynamic design. *Scientific Data*, 13 (1):1113, 2026.

Haotian Ye, Haowei Lin, Jiaqi Han, Minkai Xu, Sheng Liu, Yitao Liang, Jianzhu Ma, James Zou, and Stefano Ermon. Tfg: Unified training-free guidance for diffusion models. *Advances in Neural Information Processing Systems*, 37:22370–22417, 2024.

Yanxuan Zhao, Peng Zhang, Guopeng Sun, Zhigong Yang, Jianqiang Chen, and Yueqing Wang. Ccdpm: A continuous conditional diffusion probabilistic model for inverse design. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17033–17041, 2024.

A Dataset Details

Composition. The SuperWing dataset (Yang et al., 2026) comprises 4,239 parameterized wing geometries and 28,856 RANS flow field solutions. Each wing is generated from scratch using a 9th-order Class-Shape Transformation (CST) airfoil, with spanwise-varying thickness, camber, twist, and dihedral prescribed by cubic spline interpolation over five control points. The planform is controlled by the leading-edge sweep angle, aspect ratio, taper ratio, kink location, and root adjustment ratio. Eight operating conditions are sampled per geometry, with Mach number in $[0.75, 0.90]$, angle of attack in $[2^\circ, 12^\circ]$, Reynolds number fixed at 20 million, and freestream temperature at 300 K. All flow fields are computed by the ADflow solver (Mader et al., 2020) with an approximate Newton–Krylov method, and only steady-state converged cases are included.

Surface representation. The simulation mesh is interpolated to a reference grid of 256×128, where the spanwise direction uses 128 evenly spaced cross-sectional planes (tips excluded) and the chordwise direction uses a fixed set of normalized positions for the upper and lower surfaces. The surface flow data includes the pressure coefficient C_p and the skin-friction coefficients, decomposed into a streamwise part $C_{f,\tau}$ and a spanwise part $C_{f,z}$, all stored at cell centers.

From surface fields to lift and drag. The lift and drag coefficients are obtained by integrating the surface pressure and skin-friction distributions over the wing surface:

$$[C_L, C_D] = R_\alpha \cdot C_F, \quad C_F = \sum_{i=0}^{N_{\text{cell}}} C_{p,i}\, \mathbf{n}_i\, A_i + (\mathbf{C}_{f,i} - \mathbf{C}_{f,i} \cdot \mathbf{n}_i)\, A_i, \tag{19}$$

where R_α is the rotation matrix for the angle of attack, and $\mathbf{n}_i$ and A_i are the normal vector and area of the i-th cell.

B FFD Parameterization Details

We parameterize each wing geometry using FFD (Sederberg & Parry, 1986) with a control lattice of size $9 \times 2 \times 11$ in the streamwise, normal, and spanwise directions, and B-spline degrees $(3, 1, 3)$, yielding 594 degrees of freedom. The base shape is the mean of all training geometries, and the lattice bounding box is obtained from the base shape with 5% padding along each direction.

For each training geometry, the control point displacements are obtained by Tikhonov-regularized least-squares fitting:

$$\boldsymbol{\delta}^* = (A^\top A + \lambda I)^{-1} A^\top (\mathcal{G} - \mathcal{G}_{\text{base}}), \tag{20}$$

where A is the B-spline basis matrix, $\mathcal{G}$ is the target geometry, $\mathcal{G}_{\text{base}}$ is the base shape, and $\lambda = 0.01$. The fitted displacements are normalized to zero mean and unit variance along each coordinate.

The normalized displacement field is reshaped into a tensor of shape $(1, 6, 9, 11)$. The six channels correspond to the two control point layers in the normal direction of the FFD lattice (the second dimension of the $9 \times 2 \times 11$ grid), each contributing three displacement components. We place the normal dimension in the channel axis rather than in the spatial axes because the two control point layers correspond to the upper and lower surfaces of the wing, which are not spatially adjacent to each other; treating them as neighboring cells in a 2D convolution would introduce spurious correlations across the two surfaces. Keeping the streamwise and spanwise axes as the spatial grid allows the 2D convolutional layers of the diffusion model to exploit local correlations among neighboring control points on the same surface. This tensor is the latent variable of the diffusion model. At inference time, the decoder inverts the normalization and applies the B-spline interpolation to the base shape.

To assess the fitting accuracy, we evaluated the FFD reconstruction on 100 randomly sampled geometries. The average mean absolute error (MAE) is 0.00063 m, corresponding to 0.056% of the mean chord (1.1266 m), and the average maximum error is 0.0047 m (0.42% of the mean chord). These errors are small relative to the geometric scales involved in the design tasks.

C Surrogate Model Details

C.1 Architecture

The surrogate model is a Transolver network (Wu et al., 2024). It takes as input the surface center points of a wing geometry, together with an operating condition $(\mathrm{AoA}, \mathrm{Ma})$, and outputs the pressure coefficient C_p and the two skin-friction components $C_{f,\tau}$ and $C_{f,z}$ on the same surface points. These fields are integrated to obtain C_L and C_D. The network is fully differentiable, so gradients can be propagated from the aerodynamic coefficients back to the geometry and then to the latent variable. The model consists of 25 Transformer layers with a hidden dimension of 256, 8 attention heads and an MLP ratio of 4. Training details are provided in Appendix E.3.

C.2 Evaluation

On the held-out test set, the average relative L_2 error across the three output fields is 4.4%. This is small relative to the range of the coefficients and is sufficient for the guidance signal used in AMG and the baselines.

C.3 Data Regeneration for Conditional Baselines

The conditional baselines are trained on FFD-parameterized shapes, but their conditioning targets are the aerodynamic coefficients of those shapes. To prevent the FFD parameterization error and the surrogate model error from affecting these baselines, we regenerate the training data as follows. For each FFD-parameterized shape, we decode it back to a geometry using the FFD decoder, evaluate the resulting geometry with the trained surrogate model at all operating conditions, and use the predicted C_L and C_D as the conditioning targets. This ensures that the conditional baselines are trained on the same FFD representation and the same surrogate model that are used at inference time, so that their performance is not affected by inconsistencies between the training and evaluation pipelines.

D Baseline Implementation Details

D.1 Diffusion Backbone

All methods in this work, including AMG and every baseline, share the same diffusion backbone. The model is a denoising network that supports both conditional and unconditional inputs: when a condition vector is provided, the condition encoder is activated and its embedding is added to the time embedding before being injected into the residual blocks; when no condition is provided, the condition encoder is skipped entirely, and the network operates as an unconditional denoiser. This design allows the model to be trained jointly in both conditional and unconditional modes, which allows the same model to be used in both conditional and guidance-based baselines.

The architecture consists of the following components. A sinusoidal position embedding encodes the time step into a 128-dimensional vector, which is then passed through a two-layer MLP with a GELU activation. A separate condition encoder maps a 4-dimensional condition vector $(\mathrm{AoA}, \mathrm{Ma}, C_L, C_D)$ into the same 128-dimensional space through a two-layer MLP with SiLU activations. The noisy input of shape $(6, 9, 11)$ is first projected by a 3×3 convolution into a hidden dimension of 128. It then passes through three residual blocks: the first expands the hidden dimension from 128 to 256, the second keeps it at 256, and the third reduces it back to 128. Within each residual block, the first convolution block applies adaptive group normalization (AdaGN), where the scale and shift are computed from the sum of the time embedding and the condition embedding. At the bottleneck, a multi-head self-attention layer with 4 heads of dimension 32 operates on the 256-dimensional feature map. The output is produced by a 1×1 convolution that maps the hidden dimension back to 6 channels. The total number of trainable parameters is approximately 3.1M. Training details are provided in Appendix E.2.

D.2 CcDPM

CcDPM (Zhao et al., 2024) is the first work to introduce diffusion models into engineering inverse design and the first to apply Vicinal Risk Minimization (VRM) to diffusion models. It also proposes

a multi-point design sampling scheme that allows the model to accept multiple operating conditions as input even though it is trained on a single condition.

We use the training method described in the original paper, including the vicinal loss and the k-d tree acceleration. At inference time, we use the multi-point design sampling scheme with guiding scales s. The guiding scales s are tuned on the first task and fixed for the rest.

D.3 AIRFOIL-DDPM

Airfoil-DDPM (Wen et al., 2025) is a conditional DDPM for multi-objective airfoil design. Its key contribution is a distance-adaptive weighting scheme that combines the noise predictions from different operating conditions based on their mutual distances in the noise space.

Airfoil-DDPM does not introduce a new training method; we therefore train it using the same unified training procedure described in Section D.1. At inference time, we use the distance-adaptive weighting scheme, with a CFG strength denoted by γ. The value of γ is selected on the first task and fixed for the rest.

D.4 STANDARD CLASSIFIER GUIDANCE (CG)

Standard CG (Dhariwal & Nichol, 2021) adds the gradient of a scalar objective to the noise prediction with a fixed guidance scale. Since the surrogate model outputs a vector (C_L, C_D) rather than a scalar log-likelihood, we define the objective as the sum of squared residuals over all targets:

$$L = \sum_{i=1}^{n} \left[(\hat{C}_{L,i} - C_{L,i}^{\text{tgt}})^2 + (\hat{C}_{D,i} - C_{D,i}^{\text{tgt}})^2 \right]. \tag{21}$$

The residual direction is $\hat{C} - C^{\text{tgt}}$, and the time-dependent factor $\sqrt{1-\bar{\alpha}_t}$ is kept as in the original formulation:

$$\hat{\epsilon}_\theta(x_t, t) = \epsilon_\theta(x_t, t) - \lambda\sqrt{1-\bar{\alpha}_t}\,\nabla_{x_t} L. \tag{22}$$

Standard CG does not introduce a new training method; we therefore train it using the same unified training procedure described in Section D.1. At inference time, we replace the classifier gradient with the surrogate-based gradient computed through the FFD decoder. The guidance scale λ is selected on the first task and fixed for the rest.

D.5 NRCG

NRCG (Vaeth et al., 2025) improves CG by evaluating the guidance signal on a one-step denoised estimate $\hat{x}_0$ rather than on the noisy latent x_t, and by smoothing the gradient across adjacent steps. We use the same scalar objective and residual direction as in standard CG, and following the original paper, the update is

$$x_{t-1} = \mu_\theta(x_t) - \lambda\beta_t\,\nu(\nabla_{x_t} L), \tag{23}$$

where $\nu(\cdot)$ is a stabilizer. We use the exponential moving average (EMA) stabilizer with $\beta_{\text{EMA}} = 0.99$, which is the best setting reported in the original paper. The guidance scale λ is selected on the first task and fixed for the rest.

D.6 DFLOW-SUR

Dflow-SUR (Yang et al., 2025) is a flow-matching framework that backpropagates the physical loss through the entire generation trajectory to optimize the initial noise. We use the same squared-error objective as above, and optimize the initial source x_0 with $K = 20$ iterations of gradient descent, and a 50-step ODE solver. The step size is selected on the first task and fixed for the rest.

D.7 HYPERPARAMETER SELECTION

AMG does not require per-task hyperparameter tuning. For each baseline, the guidance weight is selected once on the first task and then fixed for all remaining tasks. The initial candidate set for each method is taken from the values recommended in its original paper or implementation. We

Table 4: Selected guidance weight for each method. The quantity is referred to as the guidance weight throughout, but its actual name and meaning differ across methods, so the values are not directly comparable.

Method	Actual name	Selected value
Standard CG	guidance scale λ	200
NRCG	guidance scale λ	50
CcDPM	guiding scale s	0.50
Airfoil-DDPM	CFG strength γ	1.2
Dflow-SUR	step size η	20
AMG (ours)	no per-task tuning	–

use a trend-guided incremental search: we evaluate each candidate on the first task and extend the search in the direction of decreasing error until the improvement becomes marginal. The number of evaluated configurations is comparable across methods.

We note that, for a generative design task, the optimal hyperparameters can in principle differ from one task to another, and a fully per-task tuning procedure would allow each baseline to use its best configuration on each task. Such a procedure is prohibitively expensive, since it would require a separate search for every task and every method. We therefore tune each baseline on the first task only and fix the selected value for the remaining tasks.

Table 4 lists the selected guidance weight for each method. We refer to this quantity as the guidance weight throughout, although its actual name and meaning differ across methods: it is a classifier guidance scale for CG and NRCG, a CFG strength for CcDPM and Airfoil-DDPM, and a step size of the initial-noise optimization for Dflow-SUR. Because these quantities play different roles in their respective formulations, their numerical values are not directly comparable. The table is provided solely to support reproduction.

E Implementation Details

E.1 Hardware

All training and inference experiments were conducted on a single compute node with 8 NVIDIA RTX 4080 Super GPUs, each with 16 GB of memory. Both the diffusion model and the surrogate model were trained on this node, using all 8 GPUs. Inference experiments were run on a single RTX 4080 Super GPU.

E.2 Diffusion Model Training

The diffusion model uses the AdaGN-based ResNet architecture described in Appendix D.1, with a hidden dimension of 128, a time embedding dimension of 128, 4 attention heads of dimension 32, and approximately 3.1M trainable parameters. It is trained on the normalized FFD displacement fields with the DDPM noise schedule, using $\beta_1 = 10^{-4}$, $\beta_T = 0.02$, and $T = 1000$ linear steps. During training, the condition is randomly dropped with probability 0.15, so that the model learns to denoise both with and without the condition. The optimizer is AdamW with a learning rate of 1×10^{-4}, weight decay 1×10^{-4}, and a cosine schedule with warmup of 100 epochs. The batch size is 64, and the model is trained for 1000 epochs. Training took approximately 1 hour on 8 RTX 4080 Super GPUs.

E.3 Surrogate Model Training

The surrogate model is the Transolver network described in Appendix C, with 25 Transformer layers, a hidden dimension of 256, 8 attention heads, an MLP ratio of 4, and a dropout rate of 0.1. It is trained with the AdamW optimizer, a learning rate of 1×10^{-4}, weight decay 1×10^{-4}, a cosine learning rate schedule with warmup of 3000 iterations, and a batch size of 2 per GPU. The training set and test set are split by geometry at a ratio of 9:1, and the same split is used for all reported results. The model is trained for 50 epochs, on 8 RTX 4080 Super GPUs, for approximately 6 hours.

Table 5: Total inference time per design task on a single RTX 4080 Super GPU.

Method	Total time
CcDPM	∼4.9 min
Airfoil-DDPM	∼5.1 min
Standard CG	∼20.5 min
NRCG	∼20.9 min
Dflow-SUR	∼45 s
AMG (ours)	∼20.7 min

E.4 Inference Cost

All inference experiments were run on a single RTX 4080 Super GPU. Table 5 reports the total inference time per design task. The differences mainly come from the generation paradigm: CFG-based methods (CcDPM, Airfoil-DDPM) are faster, CG-based methods (Standard CG, NRCG) are slower because they compute a guidance gradient at each denoising step, and Dflow-SUR uses a flow-matching formulation with fewer sampling steps and is the fastest. AMG introduces additional computation, including FFD decoding, surrogate forward and backward passes, and a 4×4 pseudo-inverse, but its total time is comparable to standard CG. This indicates that the overhead introduced by AMG is not significant.

E.5 Pseudo-inverse Configuration

The linear system $J_t \Delta x_t \approx \mathbf{b}^{(t)}$ is solved using the Moore–Penrose pseudo-inverse via `torch.linalg.pinv` with the default tolerance. No Tikhonov regularization is applied. The matrix $J_t J_t^\top$ has size $2n \times 2n$, which is 4×4 for $n = 2$. All computations are performed in `float32`.

We use the pseudo-inverse rather than a direct solve because $J_t J_t^\top$ can be numerically singular when two rows of J_t are nearly collinear. In that case, the pseudo-inverse returns the minimum-norm least-squares solution, which is the intended behavior: conflicting targets should not be amplified into a large update. Because the system is only 4×4, the internal SVD is negligible compared with the surrogate forward and backward passes, and we observed no solver failure in any of our runs.

F Design Tasks

We construct 16 multi-point design tasks by partitioning the operating range of the SuperWing dataset into 5 intervals along the angle of attack and 5 intervals along the Mach number, yielding 25 operating-condition bins. For each bin, we compute the mean and standard deviation of the corresponding C_L and C_D values in the dataset. A design point is generated by sampling $(\mathrm{AoA}, \mathrm{Ma})$ uniformly at random from the dataset range, identifying the bin that contains it, and then sampling C_L and C_D uniformly from within one standard deviation of the bin mean. Each task consists of $n = 2$ such design points, which must be satisfied by a single geometry. The resulting 16 tasks are listed in Table 6.

G Best-Step, Final-Step, and MAPE Evolution

Table 7 reports the average best-step and final-step MAPE of all methods over the 16 tasks, together with the average step at which the best MAPE is achieved. For the diffusion-based methods, the step is a denoising step between 0 and 1000, with 0 being the final step. For Dflow-SUR, the step is an initial-noise optimization iteration, and each iteration contains 50 ODE steps; we report it in iter steps to match the horizontal axis of Figure 2. For every baseline, the final-step MAPE is higher than the best-step MAPE, indicating that the sampling trajectory drifts away from the target in the late stage. AMG achieves the lowest values in both cases, and its best-step MAPE (4.32%) is even lower than the final-step MAPE of every baseline.

Table 6: The 16 multi-point design tasks used in our experiments. Each task consists of two design points, and a single geometry must satisfy both.

	Point 0				Point 1			
Task	AoA	Ma	C_L^{tgt}	C_D^{tgt}	AoA	Ma	C_L^{tgt}	C_D^{tgt}
0	5.96	0.764	0.344	0.0119	10.00	0.754	0.787	0.0306
1	2.39	0.821	-0.029	0.0167	4.46	0.851	0.084	0.0156
2	2.43	0.753	-0.086	0.0128	10.72	0.758	0.939	0.0511
3	6.87	0.824	0.517	0.0225	11.10	0.770	0.926	0.0576
4	3.91	0.785	0.022	0.0137	9.27	0.868	0.728	0.0429
5	5.55	0.887	0.194	0.0228	2.13	0.809	0.056	0.0118
6	6.50	0.759	0.471	0.0171	2.38	0.880	-0.113	0.0280
7	4.81	0.865	0.184	0.0233	9.02	0.768	0.743	0.0318
8	7.83	0.870	0.326	0.0279	8.81	0.845	0.587	0.0477
9	3.53	0.872	-0.193	0.0218	7.39	0.788	0.491	0.0243
10	7.12	0.894	0.363	0.0238	6.16	0.851	0.422	0.0346
11	11.63	0.887	0.789	0.1046	5.60	0.831	0.320	0.0130
12	4.89	0.871	0.103	0.0332	10.28	0.886	0.830	0.0980
13	2.99	0.812	0.084	0.0224	11.11	0.851	0.765	0.0513
14	6.64	0.879	0.345	0.0322	6.30	0.851	0.536	0.0304
15	5.35	0.813	0.286	0.0135	9.46	0.805	0.597	0.0422

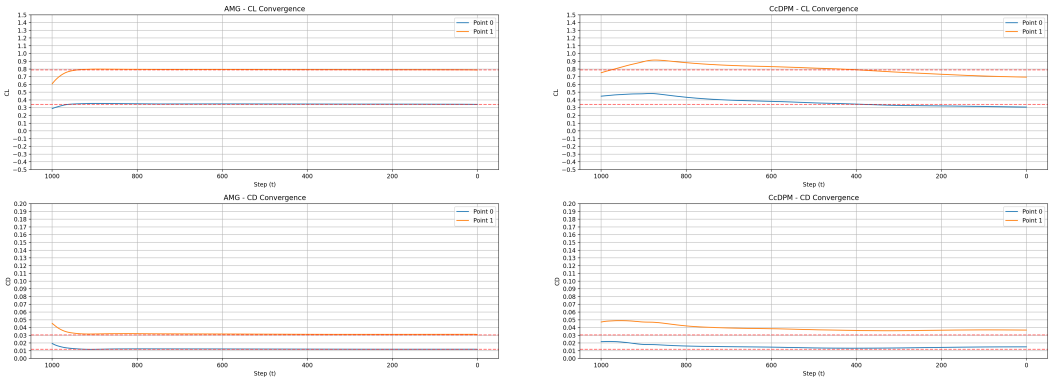
Table 7: Average best-step and final-step MAPE (%) over 16 tasks, and the average step at which the best MAPE is achieved. For the diffusion-based methods, the best step is a denoising step. For Dflow-SUR, it is an iter step; each iter step contains 50 ODE steps, so the values are not directly comparable with the denoising steps of the other methods.

Method	Best-step MAPE (%)	Final-step MAPE (%)	Avg. best step
CcDPM	27.61	33.10	469.88
Airfoil-DDPM	26.81	34.37	453.56
Standard CG	14.03	40.22	517.38
NRCG	22.76	26.08	291.19
Dflow-SUR	18.63	31.97	8.69 iter
AMG (ours)	4.32	9.24	149.25

Figure 2 shows the evolution of C_L and C_D along the generation trajectory for all six methods on task 0. The curves stay close to their target values over the middle of the trajectory and drift away near the final steps, which is why the final-step MAPE is not used as the primary metric in the main text. Because the C_L and C_D values remain close to their targets over a long middle segment, the step at which the best MAPE is attained is sensitive to small numerical fluctuations and varies across tasks, so we do not treat the best step as a quantity for comparing methods.

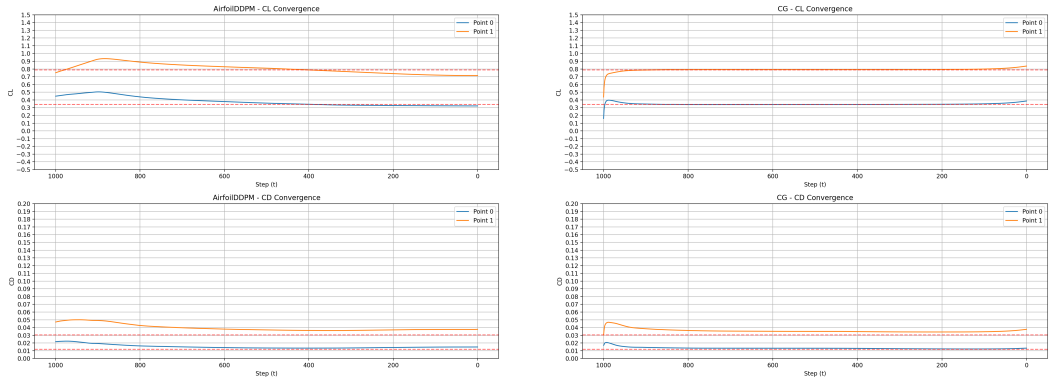
H Generated Geometries in Experiments

We show the generated geometry of each method on all 16 multi-point design tasks, taken at the step where the MAPE is minimized during sampling. Each figure is a 4×4 grid arranged in row-major order: the first row shows tasks 0–3 from left to right, the second row shows tasks 4–7, and so on. Colors encode the vertical coordinate z of the surface and are used only for visualization. For each figure, the tasks that are visually identified as invalid are listed in the caption. We note that these invalid geometries do not exhibit issues such as self-intersection or zero thickness; they are identified as invalid only because of subjectively judged shape distortion, and are therefore difficult to detect using quantitative methods.



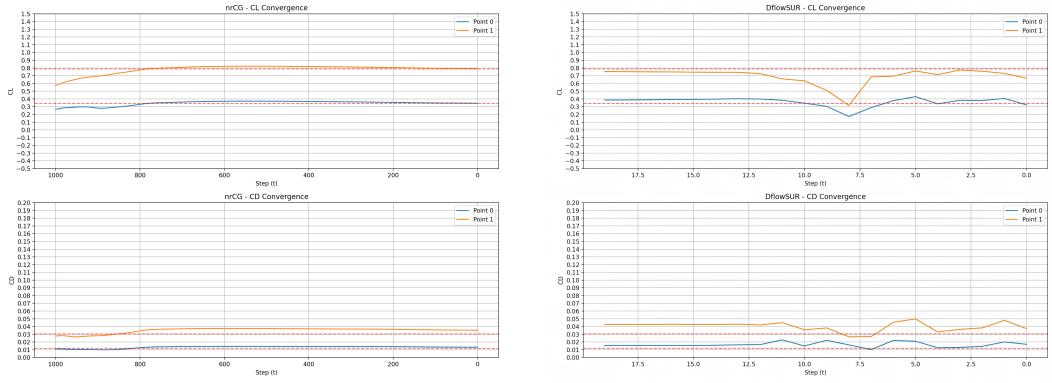
(a) AMG

(b) CcDPM



(c) Airfoil-DDPM

(d) Standard CG



(e) NRCG

(f) Dflow-SUR

Figure 2: Evolution of C_L and C_D along the generation trajectory for all six methods on task 0. For the diffusion-based methods (AMG, CcDPM, Airfoil-DDPM, Standard CG, NRCG), the horizontal axis is the denoising step t. For Dflow-SUR, the horizontal axis is the initial-noise optimization iter step, from 0 to 20; each iter step contains 50 ODE steps.



Figure 3: Best generated geometries of CcDPM on the 16 tasks. Invalid tasks: None.



Figure 4: Best generated geometries of Airfoil-DDPM on the 16 tasks. Invalid tasks: None.



Figure 5: Best generated geometries of standard CG on the 16 tasks. Invalid tasks: 13.



Figure 6: Best generated geometries of NRCG on the 16 tasks. Invalid tasks: 4.

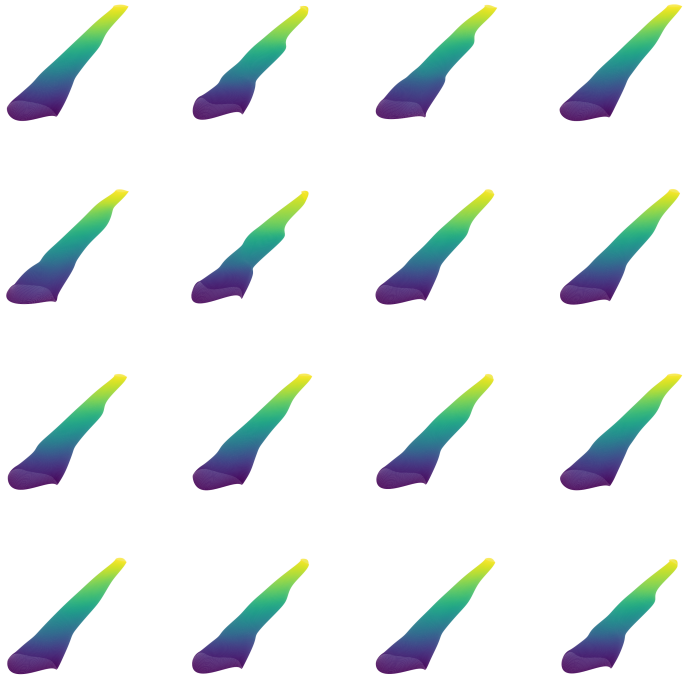


Figure 7: Best generated geometries of Dflow-SUR on the 16 tasks. Invalid tasks: 1, 2, 4, 5, 8, 9, 13, 15.



Figure 8: Best generated geometries of AMG on the 16 tasks. Invalid tasks: 10.



Figure 9: Best generated geometries of the fixed-weight variant on the 16 tasks. Invalid tasks: 15.



Figure 10: Best generated geometries of the no-scaling variant on the 16 tasks. Invalid tasks: 1, 8, 10, 12, 13, 14.



Figure 11: Best generated geometries of the uniform-scaling ($1/t$) variant on the 16 tasks. Invalid tasks: 15.