From Positive to Negative: On the Role of Negative Data in Enhancing Generative Models for Engineering Constraint Satisfaction

Lyle Regenwetter[0000−0002−8769−0037] and Faez Ahmed[0000−0002−5227−2628]

Massachusetts Institute of Technology, Cambridge MA 02139, USA
Design Computation and Digital Engineering Lab
{regenwet, faez}@mit.edu
http://decode.mit.edu

Abstract. Generative Artificial Intelligence has the potential to transform engineering sectors by enhancing design innovation and automating processes. However, despite advances in their data, training, and architectures, generative models still struggle to effectively and reliably satisfy constraints. This shortcoming presents a significant challenge with their adoption in engineering design tasks, where design constraints are ubiquitous. This difficulty is rooted in the similarity-based training objective of generative AI models, in which they learn to mimic the statistical distribution of a dataset of constraint-satisfying examples (positive data). We assert that generative models can be more effectively trained by examining constraint-violating examples (negative data) in addition to positive data. These "Negative Data Generative Models" (NDGMs) can thereby learn to avoid sampling from constraint-violating regions of the sample space. To demonstrate this principle, we propose a type of NDGM, then benchmark this formulation against vanilla models on two 2D test problems and two engineering design problems related to gearbox and concrete beam design. We showcase that NDGMs achieve significantly (2-30x) better constraint satisfaction compared to vanilla generative models. Moreover, they learn these constraints with only a fraction of the training data compared to vanilla generative models. Since NDGMs require only a handful of example to adjust their learned densities, they are significantly more agile and adaptable than vanilla generative models and may be much more effective in continuous data streams as seen in Dynamic Data-Driven Application Systems (DDDAS). Our findings suggest that NDGMs could play a crucial role in overcoming the constraints satisfaction challenges in current generative models, thereby broadening the scope and applicability of generative AI in critical engineering domains.

Keywords: Generative AI · Negative Data · Design Constraints

1 Introduction

Generative AI models have achieved remarkable success in various fields such as vision, language, and speech. Nonetheless, despite the availability of extensive datasets, these models often struggle with precision, producing samples that are physically impossible, factually incorrect, or otherwise "invalid," such as unbalanced objects in natural scenes, anatomical errors, and inaccurate text responses. This issue of invalidity can be understood as a type of constraint violation — ideally, generative models would be restricted to producing only valid outputs. Constraint violation is critically important in fields like engineering design, where high-stakes (including safety-critical) constraints are involved. For instance, mechanical components must often adhere to geometric constraints (such as avoiding disconnected or colliding parts), functional requirements (such as load-bearing capabilities or weight limits), industry standards, and manufacturing constraints. This makes the application of generative models to design and engineering tasks challenging. Despite this, the usage of generative models in design has increased exponentially in recent years. However, the challenge of constraint violation in generative models for design remains largely unsolved.

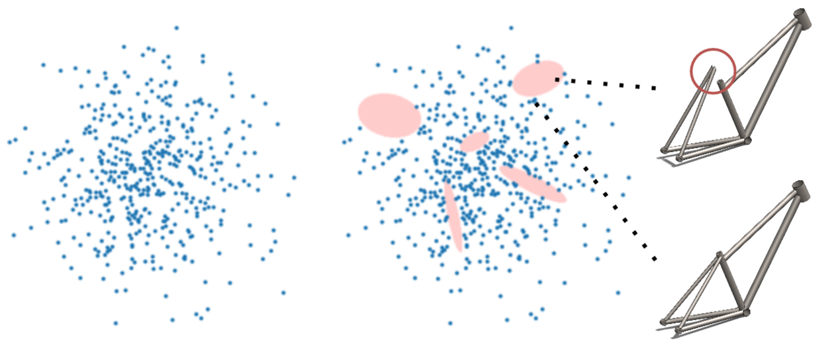


Fig. 1: Generative models are classically trained to learn a probability distribution over "positive" (constraint-satisfying) data. We show how learning constraints using this training formulation is difficult. What a generative model may see as a simple 2D Gaussian distribution (left) may in fact be a complicated distribution riddled with gaps in its support caused by unknown constraints (right image showing constraint-violating regions with red shaded areas). As constraint-violating designs may be very similar to constraint-satisfying designs, learning only from positive examples can mislead generative models.

Generative models are classically trained using only constraint-satisfying datapoints during training, which we call "positive" data. This positive data varies widely based on the application, encompassing everything from smooth airfoils and optimal structural designs to functional metamaterial unit cells. By examining the distribution of constraint-satisfying examples, they attempt to mimic this distribution and generate samples that are, ideally, constraint-satisfying as well. Using this training strategy, the generative model never sees any examples of constraint-violating datapoints, which we call "negative" data. Learning the constraint boundary with this approach is practically impossible with limited data, as the model effectively needs to learn a binary classification task using only examples of one class. By instead studying both positive and negative data during training, we assert that generative AI models can better learn to satisfy constraints. In this training setting, the generative AI model roughly attempts to generate samples that are probable according to its estimate of the positive data distribution and improbable according to its estimate of the negative data distribution. Since positive and negative data are disjoint, this training formulation is still consistent with the classic distribution-matching objective [21] because negative datapoints should have zero density in the original positive distribution that the model is trying to mimic. We call models trained under this new formulation "Negative Data Generative Models", or "NDGMs."

By leveraging examples of what not to generate, NDGMs are more precise, adaptable, and data-efficient than classic genrative models. This makes them particularly effective in continuous data streams where the generative model must quickly adapt to changing conditions reflected in only a few new datapoints. In this paper, we demonstrate that the NDGM training formulation can significantly improve constraint satisfaction in engineering design tasks and can often do so with significantly less data than the conventional training approach for generative AI models.

2 Background

2.1 Generative AI

Generative AI models are conventionally trained to maximize the statistical similarity between the samples they generate and their training dataset. We refer to the positive data distribution and the distribution sampled by the generative model as $p_p(x)$ and $p_\theta(x)$, respectively. Thus, the goal of the generative model is to find an optimal setting, θ^*, for its internal weights, θ, such that $p_\theta^* \approx p_p$. The similarity of p_θ^* and p_p is measured with a discrepancy measure, such as Kullback–Leibler (KL) divergence:

$$\mathbb{KL}[p_\theta \| p_p] = \int p_\theta(\mathbf{x}) \left[\log \frac{p_\theta(\mathbf{x})}{p_p(\mathbf{x})} \right] d\mathbf{x}. \tag{1}$$

To find θ^*, we minimize the discrepancy. Therefore, the generative model effectively searches for the solution to the following optimization problem:

$$\theta^* = \arg\min_\theta \mathbb{KL}[p_\theta \| p_p]. \tag{2}$$

However, since Equation 2 is intractable to directly optimize, a tractable variant of Equation 2 is frequently used for generative model training [11, 12] instead. Alternatively, plug-in or direct estimators of the divergence measure can be used in the optimization objective as a stand-in for the exact divergence [10, 18, 24–26]. In either case, as the number of data samples, N, approaches infinity, θ approaches θ^* ($N \to \infty \;\Rightarrow\; \theta \to \theta^*$). Naturally, N is finite in practice, implying that a finite discrepancy between θ^* and θ will remain. This discrepancy is particularly pronounced in data-limited domains such as engineering design, leading to models generating samples that do not satisfy constraints.

2.2 Constraint Satisfaction in Design

Engineering design synthesis is inherently subject to numerous constraints, such as geometric limitations, performance criteria, safety regulations, and engineering standards. The ubiquity of design constraints extends to many problems where practitioners have applied generative AI models [3–6, 15, 17]. Despite this, generative models often struggle to adhere to these constraints [21, 28], and there are few broad solutions to address the significant issues of constraint violation observed in designs generated by generative AI [20]. Nonetheless, there are ample resources for improving constraint satisfaction. Many engineering datasets [2, 8, 13, 22] contain examples of constraint-violating designs and several more contain constraint checks [27, 29]. Occasionally, negative data—instances where design criteria are not met—can be obtained at no additional cost. For example, during the generation of datasets where positive examples are selected through a process that rejects and discards non-compliant samples [2, 22], this discarded data can be retained and used to enhance the training of generative models. Despite the availability of negative data on which to train NDGMs, NDGMs remain underexplored in engineering design applications.

3 Negative Data Generative Models

In this paper, we will propose a variant of a commonly used generative modeling framework, the Generative Adversarial Network (GAN) [1, 10, 14, 16, 25]. GANs are powerful learners capable of generating both diverse and realistic samples. Although we will modify the GAN to consider negative data, we first introduce the standard GAN. Within this paper, we refer to these vanilla generative models as "VGMs." A GAN is comprised of two models: a generator f_θ, which samples according to a distribution p_θ and learns to approximate the density p_p, and a discriminator f_ϕ, which is a binary classifier that learns to distinguish samples from p_θ and p_p. The standard training objective can be written as:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{p_p(\mathbf{x})}[\log f_\phi(\mathbf{x})] + \mathbb{E}_{p_\theta(\mathbf{x})}[1 - \log(f_\phi(\mathbf{x}_\theta))] \tag{3}$$

The GAN is trained by iterating over $\min_\theta \max_\phi \mathcal{L}(\theta, \phi)$. Given infinite samples from p_p, the optimal discriminator prediction and generator distribution are:

$$f_\phi(\mathbf{x}) = \frac{p_p(\mathbf{x})}{(p_\theta(\mathbf{x}) + p_p(\mathbf{x}))},\ p^*_\theta(\mathbf{x}) = p_p(\mathbf{x}). \tag{4}$$

To augment the GAN with negative data, we now seek to guide training using a distribution of negative data, p_n, which we assume is disjoint from p_p. We do so by pretraining a second discriminative model, ψ to distinguish p_p and p_n. If perfectly trained, it will learn:

$$f_\psi(\mathbf{x}) = \frac{p_p(\mathbf{x})}{(p_n(\mathbf{x}) + p_p(\mathbf{x}))} \tag{5}$$

After pretraining this additional discriminative model, it is incorporated into the GAN's loss term as follows:

$$\begin{aligned} \mathcal{L}(\theta, \phi) = &\mathbb{E}_{p_p(\mathbf{x})}[\log f_\phi(\mathbf{x})] + \mathbb{E}_{p_\theta(\mathbf{x})}[1 - \log(f_\phi(\mathbf{x}_\theta))] \\ &+ \alpha\, \mathbb{E}_{p_\theta(\mathbf{x})}[1 - \log(f_\psi(\mathbf{x}_\theta))] \end{aligned} \tag{6}$$

Here, α is a term that modulates the contribution of the negative data to the overall learning objective. NDGMs have seen preliminary use in engineering design [9, 19], but are still underexplored.

4 Experiments

In this section, we dive into a performance analysis of NDGMs as compared to VGMS on a variety of 2D and engineering problems. We show that NDGMs are more effective at satisfying constraints and are more data-efficient than VGMs.

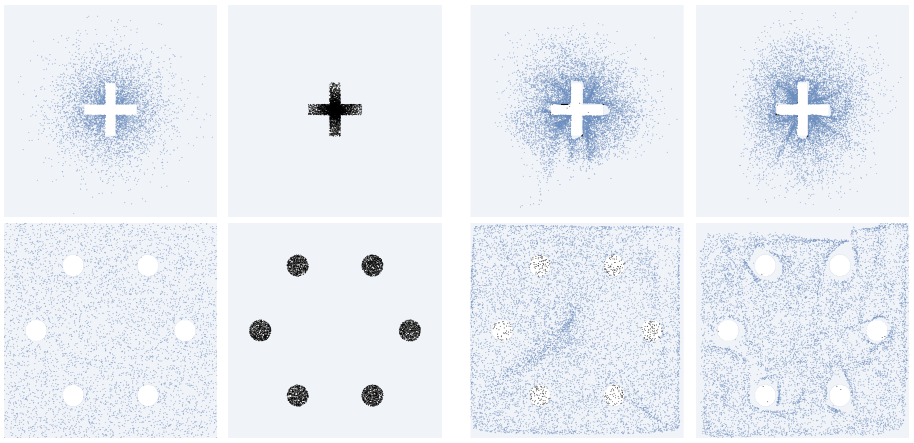
4.1 Visualizing NDGMs on 2D Densities

We first evaluate our NDGM architecture on a couple of easy-to-visualize 2D test problems. The first problem consists of a standard 2D Gaussian distribution with a plus-shaped constraint in the center of the distribution. Any datapoint within this plus shape violates the constraint and is considered a negative datapoint. The second problem consists of a 2D uniform distribution with six small circular constraints. Once again, any datapoint within one of these regions violates constraints and is considered negative. Figures 2a and 2b illustrate the positive and negative data distributions for both `Problem 1` (top) and `Problem 2` (bottom).

Models. We train a VGM and an NDGM on each problem. The VGM is a classic generative adversarial network (GAN) and the NDGM is a classifier-augmented GAN introduced in Section 3. We train the VGM model on only the positive data (as is classic for VGMs), and we train the NDGM on both the positive and negative data.

Metrics. We score each model on three metrics. The first of these is the fraction of generated samples that violate the constraints, a metric that we call "invalidity." The second is a common distributional similarity metric known as Maximum Mean Discrepancy (MMD). Finally, we evaluate a variant of F_1 score proposed for generative models [23] as another method of measuring distributional similarity.

Results. Figures 2c and 2d visualize the generated distributions from the VGM and NDGM. Quantitative scores are included in Table 1. In `Problem 1`,



(a) Positive Data (b) Negative Data (c) VGM (Baseline) (d) NDGM (Ours)

Fig. 2: Comparison of VGM vs. NDGM on 2D densities. `Problem 1` on top, `Problem 2` on the bottom. Positive data is shown in blue, while negative data is shown in black. Similarly, valid samples are shown in blue while invalid samples are shown in black. The VGM struggles to precisely learn the constraint boundaries in `Problem 1` and almost completely fails to learn the constraints in `Problem 2`. The NDGM effectively learns constraints in both problems and avoids the areas with black samples.

the VGM largely avoids the constraint-violating region but struggles to exactly learn the constraint boundary. Therefore, it generates a fair number of points that lie just across the boundary, as well as a few scattered points in the middle of the region. In contrast, the NDGM almost perfectly learns the constraint and generates exceedingly few invalid samples. The models perform similarly in distributional similarity metrics and the VGM generates 12x as many invalid datapoints as the NDGM. In `Problem 2`, the VGM largely ignores the constraints, causing it to generate 29x as many constraint-violating samples as the NDGM, which effectively avoids the constraints. In this second problem, the NDGM not only excels in validity but also significantly outperforms the VGM in distributional similarity metrics.

4.2 Is Negative Data More Efficient than Positive Data?

Given unlimited data, it is theoretically possible for a generative model to exactly recover the original data distribution it is modeling. However, in practice, generative models are almost always data-constrained. This is particularly common in engineering and design tasks, where data limitations are notoriously constraining. Therefore, data efficiency is particularly important in generative models for engineering design applications.

In this section, we demonstrate that training generative models with negative data can be significantly more data-efficient compared to the standard training

Table 1: Quantitative scores for VGM and NDGM on 2D density problems. Three instantiations of each model are tested and their scores are averaged. Our NDGM scores significantly better in invalidity. Compared to our NDGM, the VGM generates 12x and 29x as many invalid samples in `Problem 1` and `Problem 2`, respectively. Across all distributional similarity metrics, the NDGM is better or comparable to the VGM. The best scores are bolded. Invalidity and MMD scores are minimized while F1 is maximized.

	`Problem 1`		`Problem 2`	
	VGM	NDGM	VGM	NDGM
Invalidity (%) ↓	0.93	**0.08**	4.03	**0.14**
Maximum Mean Discrepancy (MMD) ↓	**0.003**	**0.003**	0.008	**0.002**
F1 Score ↑	**0.94**	**0.94**	0.87	**0.95**

Table 2: Comparison of NDGM vs VGM performance on different amounts of data. Invalidity score (%), which is minimized (↓), is shown. Scores are averaged over 8 instantiations. NDGMs trained on data split evenly between positive and negative significantly outperform even VGMs trained on double as much data by 3-8x (relevant cells bolded).

		`Problem 1`			`Problem 2`		
Model	Negative	Positive Datapoints					
Type	Datapoints	500	1000	2000	500	1000	2000
VGM	0	1.964	1.096	**0.755**	2.855	2.092	**2.331**
NDGM	500	**0.229**	0.161	0.183	**0.285**	0.426	0.320
NDGM	1000	0.204	0.286	0.214	0.172	0.190	0.211
NDGM	2000	0.143	0.317	0.176	0.151	0.104	0.169

employed in VGMs. We test VGMs using different amounts of positive data and measure their performance. We then test NDGMS on different amounts of both positive and negative data and compare their performance to the VGMs. The models are benchmarked on both of the 2D test problems from the previous section. Table 2 displays the results of the study, showing invalidity scores over the various combinations of positive and negative data. Though some stochasticity in model performance can be observed, NDGMs clearly outperform the VGMs tested. In fact, the NDGMs trained with only 500 positive and 500 negative datapoints significantly outperform VGMs trained on 2000 positive datapoints. This marks a 3-8x improvement in constraint satisfaction using half as much data.

4.3 Constraint Satisfaction in Design Problems

Having tested NDGMs in 2D problems and showcased their data efficiency, we move on to benchmark their performance against VGMs in two select engineering design problems. We test the same metrics as in the 2D experiments. These datasets are described below:

Gearbox: This gearbox (speed-reducer) design problem, obtained from [7], features 7 parameters describing key geometric components like shaft diameters, number of teeth on gears, and face width of gears. Nine constraints are given, spanning considerations like bending stress on gear teeth, transverse stress and deflection on shafts, and surface stresses. The optimization objective is not utilized. 1000 positive samples and 1000 negative samples are selected using uniform random sampling.

Concrete Beam: Also adopted from [7], this problem centers around the design of a simply supported concrete beam under a simple distributed load case. The beam is parameterized using a cross-sectional area, base length, and height and is subject to a safety requirement indicated in the American Concrete Institute (ACI) 319-77 code. The optimization objective is not utilized to focus on constraints. 1000 positive samples and 1000 negative samples are selected using uniform random sampling.

Table 3: Performance comparison of NDGMs and VGMs on three engineering problems. Scores are averaged over at least 3 runs for each of the datasets. NDGMs and VGMs perform similarly on distributional similarity metrics (MMD, F1). However, NDGMs generate only 1/3 to 1/2 as many invalid samples as VGMS.

	`Gearbox`		`Concrete Beam`	
	VGM	NDGM	VGM	NDGM
Invalidity (%) ↓	0.57	**0.19**	1.22	**0.62**
Maximum Mean Discrepancy (MMD) ↓	**0.002**	**0.002**	**0.002**	**0.002**
F1 Score ↑	**0.94**	**0.94**	**0.96**	0.95

Results: Overall the NDGMs and VGMs perform comparably in distribution matching scores. However, our NDGMs significantly outperform the VGMs in constraint satisfaction scores. The VGM generates double (`Concrete Beam`) or triple (`Gearbox`) as many invalid designs as the NDGM. These results underline the superior capability of NDGMs in meeting stringent design constraints, setting a promising direction for their application in complex engineering tasks where reliability and precision are paramount.

5 Incorporation into Dynamic Data-Driven Application Systems

In many applications involving generative models, their objectives and constraints are constantly evolving. For example, a large language model may want to adjust and constrain responses based on news headlines or newly discovered scientific evidence. Or consider a model trained to generate possibilities for flight trajectories, which may need to constantly update the types of solutions it generates based on changing weather or congestion conditions.

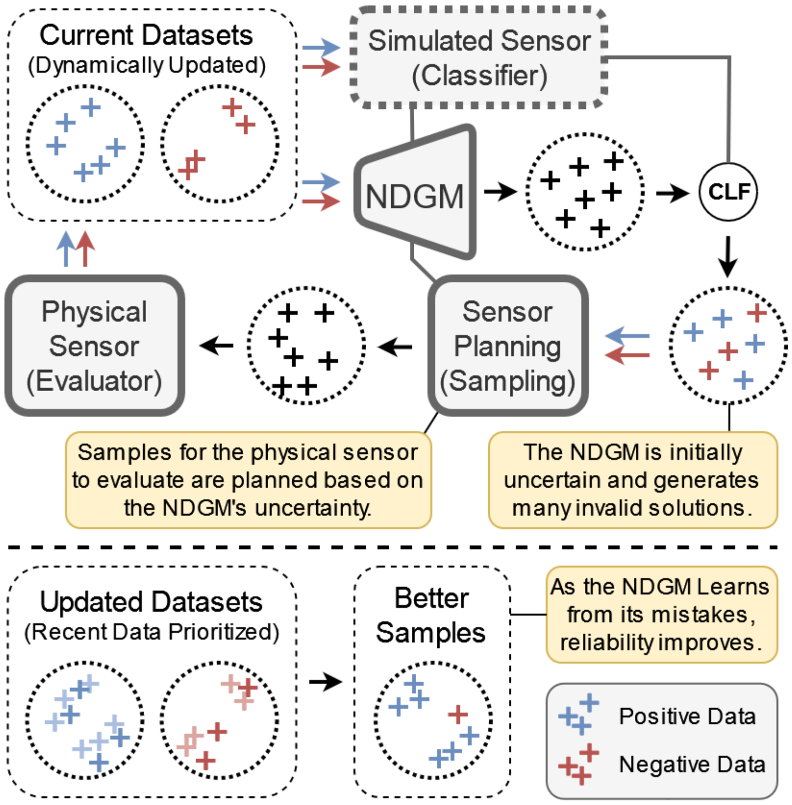

Current Datasets
(Dynamically Updated)
Simulated Sensor
(Classifier)
NDGM
CLF
Physical
Sensor
(Evaluator)
Sensor
Planning
(Sampling)
Samples for the physical sensor to evaluate are planned based on the NDGM's uncertainty.
The NDGM is initially uncertain and generates many invalid solutions.
Updated Datasets
(Recent Data Prioritized)
Better
Samples
As the NDGM Learns from its mistakes, reliability improves.
Positive Data
Negative Data

Fig. 3: Example use of NDGMs in Dynamic Data-Driven Application Systems. The NDGM can be quickly and effectively updated according to changing constraints and new data.

As we have showcased, example-based learning of constraints through negative data using NDGMs is a fast and sample-efficient method to constrain generative models. This makes NDGMs well-suited for usage in DDDAS, where changing conditions and constraints may be reflected in only a few critical examples. By avoiding constraint-violation in a targeted manner, NDGMs may be effective candidates for generative modeling in DDDAS environments. An illustration of this vision is included in Figure 3. In this scenario, the classifier, which is used to guide the generative model, fills the role of a simulated sensor. The data is dynamically updated using samples whose ground-truth validity is evaluated by a physical sensor. The points to query are sampled using a sensor planning algorithm using the uncertainty of the NDGM and the simulated sensor to guide selection of new samples.

6 Future Work and Limitations

We have demonstrated that NDGMs can be significantly more effective and efficient than VGMs. However, NDGMs rely on a source of high-quality negative data. As discussed, negative data is generally abundant in design problems and may be cheaper to generate than positive data. However, this abundance is problem-dependent. In some problems, sourcing or generating high-quality negative data may not be straightforward. Furthermore, not all negative data may be equally informative for NDGMs and more research is needed in scoring their utility or incorporating their utility within the model training. Future research may investigate generalizable methods to generate or efficiently sample negative data to train NDGMs. Though we have hypothesized that NDGMs may be effective in DDDAS, confirming the extent of their effectiveness would require specialized experiments in the DDDAS problem formulation.

7 Conclusion

In this paper, we have presented a comparative case study of "Negative Data Generative Models" (NDGMs) against vanilla generative models (VGMs). We showcased overwhelmingly superior constraint-satisfaction ability in NDGMs compared to VGMs. Crucially, NDGMs often outperform VGMs even when trained on significantly less data, making them generally more effective and more efficient. This data-efficiency and adaptability may enable their effective use in DDDAS. Though NDGMs and VGMs perform similarly in distributional similarity metrics across most test cases, NDGMs can in some cases even achieve significantly higher distribution-matching performance than VGMs. Given the overwhelming superiority of NDGMs in many engineering design tasks, we advocate for the more widespread use of NDGMs in data-driven design practice and DDDAS.

Acknowledgments. The authors thank Giorgio Giannone, Akash Srivastava, and Dan Gutfreund for assisting in the conceptual development of negative data generative models for design.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International conference on machine learning. pp. 214–223. ICML'17, PMLR, JMLR.org (2017)
2. Bagazinski, N.J., Ahmed, F.: Ship-d: Ship hull dataset for design optimization using machine learning. arXiv preprint arXiv:2305.08279 (2023)
3. Bilodeau, C., Jin, W., Jaakkola, T., Barzilay, R., Jensen, K.F.: Generative models for molecular discovery: Recent advances and challenges. Wiley Interdisciplinary Reviews: Computational Molecular Science **12**(5), e1608 (2022)
4. Chen, Q., Wang, J., Pope, P., Chen, W., Fuge, M.: Inverse design of two-dimensional airfoils using conditional generative models and surrogate log-likelihoods. Journal of Mechanical Design **144**(2), 021712 (2022)
5. Chen, W., Fuge, M.: Synthesizing designs with interpart dependencies using hierarchical generative adversarial networks. Journal of Mechanical Design **141**(11), 111403 (2019)
6. Cheng, Y., Gong, Y., Liu, Y., Song, B., Zou, Q.: Molecular design in drug discovery: a comprehensive review of deep generative models. Briefings in bioinformatics **22**(6), bbab344 (2021)
7. Gandomi, A.H., Yang, X.S.: Benchmark problems in structural optimization. In: Computational optimization, methods and algorithms, pp. 259–281. Springer (2011)
8. Giannone, G., Ahmed, F.: Diffusing the optimal topology: A generative optimization approach. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. vol. 87301, p. V03AT03A012. American Society of Mechanical Engineers (2023)
9. Giannone, G., Regenwetter, L., Srivastava, A., Gutfreund, D., Ahmed, F.: Learning from invalid data: On constraint satisfaction in generative models. arXiv preprint arXiv:2306.15166 (2023)
10. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680. NIPS'14, MIT Press, Cambridge, MA, USA (2014)
11. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 6840–6851. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf
12. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
13. Mazé, F., Ahmed, F.: Diffusion models beat gans on topology optimization. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). Washington, DC (2023), https://arxiv.org/abs/2208.09591
14. Mohamed, S., Lakshminarayanan, B.: Learning in implicit generative models. arXiv preprint arXiv:1610.03483 (2016)
15. Nie, Z., Lin, T., Jiang, H., Kara, L.B.: Topologygan: Topology optimization using generative adversarial networks based on physical fields over the initial domain. Journal of Mechanical Design **143**(3), 031715 (2021)
16. Nowozin, S., Cseke, B., Tomioka, R.: f-gan: Training generative neural samplers using variational divergence minimization. Advances in neural information processing systems **29** (2016)

17. Oh, S., Jung, Y., Kim, S., Lee, I., Kang, N.: Deep generative design: Integration of topology optimization and generative models. Journal of Mechanical Design **141**(11) (2019)
18. Poole, B., Ozair, S., Van Den Oord, A., Alemi, A., Tucker, G.: On variational bounds of mutual information. In: International Conference on Machine Learning. pp. 5171–5180. PMLR (2019)
19. Regenwetter, L., Ahmed, F.: Design target achievement index: A differentiable metric to enhance deep generative models in multi-objective inverse design. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. vol. 86236, p. V03BT03A046. American Society of Mechanical Engineers (2022)
20. Regenwetter, L., Nobari, A.H., Ahmed, F.: Deep generative models in engineering design: A review. Journal of Mechanical Design **144**(7), 071704 (2022)
21. Regenwetter, L., Srivastava, A., Gutfreund, D., Ahmed, F.: Beyond statistical similarity: Rethinking metrics for deep generative models in engineering design. arXiv preprint arXiv:2302.02913 (2023)
22. Regenwetter, L., Weaver, C., Ahmed, F.: Framed: Data-driven structural performance analysis of community-designed bicycle frames (2022)
23. Sajjadi, M.S., Bachem, O., Lucic, M., Bousquet, O., Gelly, S.: Assessing generative models via precision and recall. Advances in neural information processing systems **31** (2018)
24. Srivastava, A., Han, S., Xu, K., Rhodes, B., Gutmann, M.U.: Estimating the density ratio between distributions with high discrepancy using multinomial logistic regression. Transactions on Machine Learning Research (2023), https://openreview.net/forum?id=jM8nzUzBWr
25. Srivastava, A., Valkov, L., Russell, C., Gutmann, M.U., Sutton, C.: Veegan: Reducing mode collapse in gans using implicit variational learning. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30, pp. 3308–3318. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/44a2e0804995faf8d2e3b084a1e2db1d-Paper.pdf
26. Sugiyama, M., Suzuki, T., Kanamori, T.: Density ratio estimation in machine learning. Cambridge University Press (2012)
27. Whalen, E., Beyene, A., Mueller, C.: Simjeb: simulated jet engine bracket dataset. In: Computer Graphics Forum. vol. 40, pp. 9–17. Wiley Online Library (2021)
28. Woldseth, R.V., Aage, N., Bærentzen, J.A., Sigmund, O.: On the use of artificial neural networks in topology optimisation. Structural and Multidisciplinary Optimization **65**(10), 294 (2022)
29. Wollstadt, P., Bujny, M., Ramnath, S., Shah, J.J., Detwiler, D., Menzel, S.: Carhoods10k: An industry-grade data set for representation learning and design optimization in engineering applications. IEEE Transactions on Evolutionary Computation **26**(6), 1221–1235 (2022)

8 Itemized Updates

1. Page 1: Remove superscript for author affiliations
2. Page 1: Updated Introduction to describe the synergies of NDGMs with DDDAS-style problems.
3. Page 2: Added highlight of synergy between NDGMs and DDDAS at end of introduction.
4. Page 4: Clarified ("Gaussian" replaced with "Gaussian distribution").
5. Page 5: Removed discussion about the NDGM abbreviation, as it is dicussed before. Moved introduction of the VGM abbreviation to the beginning of the section.
6. Page 6: Edited figure reference to refer to specific subfigures rather than main figure in results section.
7. Page 8-9: Added section: "Incorporation into Dynamic Data-Driven Application Systems"
8. Page 10: Added note on future work on experiments in DDDAS problem formualtion.
9. Page 10: Mention DDDAS in conclusion.
10. Page 10: Correction ("VGM" replaced with "VGMs").
11. All Pages: Update abbreviated title