

Inference-Time Scaling of Diffusion Models via Progressive Seed Pruning

Rogério Guimarães¹ and Pietro Perona¹

California Institute of Technology, Pasadena CA 91125, USA

{rogerio,perona}@caltech.edu

<https://vision.caltech.edu>

Abstract. Diffusion and flow-matching models dominate conditional image generation, yet inference-time scaling for these models is far less developed than for autoregressive language models. Because final quality is highly sensitive to the initial noise seed, many approaches spend extra compute on seed search or resampling under a black-box reward, but typically maintaining a constant memory footprint throughout inference. We show that relaxing this constraint enables an underexplored inference-time scaling axis: by front-loading exploration, evaluating many seeds early, and pruning aggressively, we can use a fixed compute budget more effectively. *Progressive Seed Pruning* (PSP) scores intermediate denoised estimates and progressively narrows the candidate set so that only promising trajectories are fully denoised, while keeping the total number of model evaluations fixed. Across diffusion and flow-matching backbones, PSP consistently improves reward-guided selection and achieves higher GenEval scores (automated) and better human evaluation on prompt-alignment than best-of- N , importance-sampling, and tree-search baselines at matched compute. Project page: vision.caltech.edu/psp. Code: github.com/rogerioagjr/psp

Keywords: Diffusion models · flow matching · inference-time scaling · reward-guided generation · prompt alignment · particle filtering

1 Introduction

Diffusion [16, 36, 38] and Flow Matching models [25, 26] have become the default engines for conditional image and video generation. At the same time, a central lesson from large language models is that *inference-time scaling* can be as important as training-time scaling: additional compute at test time, via search, sampling, and verification, can yield large quality gains [2, 8, 40, 47, 48]. For diffusion-style generators, the dominant knob has been to increase denoising steps, but this is often an inefficient way to spend extra compute.

A growing body of work points to a different lever: for modern text-to-image systems, the *random seed* (initial noise) can strongly influence generation quality [30, 45]. This motivates inference-time strategies that treat generation as *search*

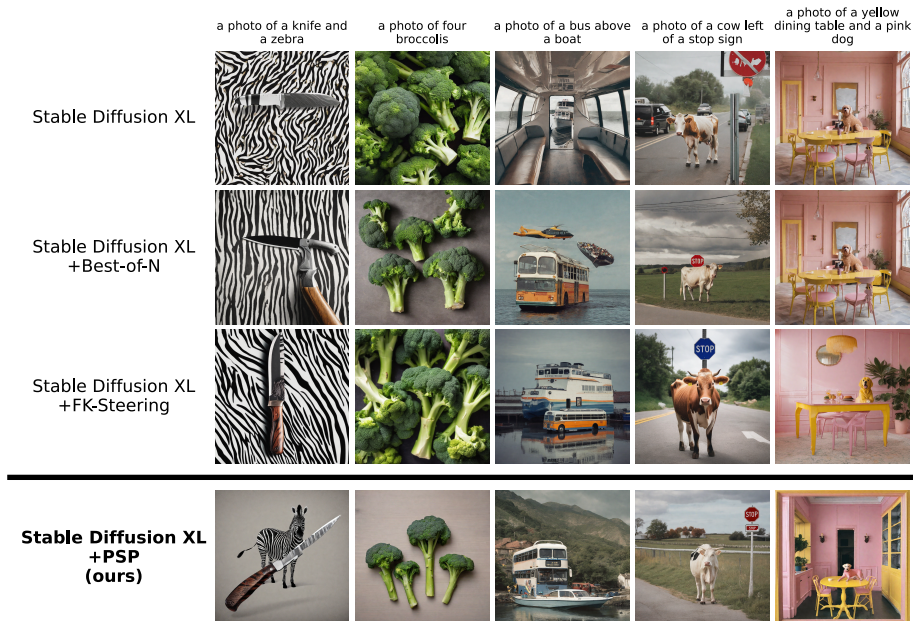


Fig. 1: Examples of image generation improvement with PSP. SDXL generations using a standard sampler and three gradient-free inference-time scaling strategies under a fixed compute multiplier $\bar{N} = 4$: BoN, FK-Steering (importance-sampling based), and our PSP. We show examples from GenEval prompts in which PSP improves on BoN and FK-Steering, satisfying prompt constraints (full results in Tab. 1).

over seeds guided by a black-box reward model. Early work showed that Best-of-N (BoN) already scales better than using more timesteps [27], but it fully denoises many candidates that are unlikely to win.

Recent methods use *intermediate* reward signals to allocate compute more efficiently, via importance sampling [34] or tree search [23, 28, 50]. These approaches are gradient-free and require no backbone finetuning, but typically enforce a *fixed* number of parallel samples (constant memory) throughout the trajectory. In contrast, particle filtering has long studied variable particle counts and early pruning [14], but this idea has not been thoroughly stress-tested for modern diffusion-style image generation.

In this paper we revisit a simple question: *if we relax the constant-memory constraint, how should we allocate compute over the denoising process?* Intermediate reward estimates often become informative relatively early. That suggests a straightforward strategy: begin with a large pool of seeds, advance them only until reward signals become meaningful, prune them progressively, and reserve the remaining denoising budget for a shrinking set of promising candidates.

We study a simple instantiation of this idea, *Progressive Seed Pruning* (PSP). PSP begins from a large pool of noise seeds, scores intermediate denoised es-

timates, and prunes on a predetermined schedule so that only promising trajectories receive the remaining denoising budget. This front-loads exploration and improves reward-guided selection over prior inference-time scaling baselines while keeping the total number of model evaluations fixed. Using a fixed schedule also aligns with deployment: predetermined prune points and survivor counts make each interval’s memory footprint and runtime predictable, facilitating allocation on its ideal setting: multi-GPU servers with elastic. Taken together, our results suggest that *variable particle populations* are a simple, general, and deployment-friendly design principle for inference-time scaling in modern generative models.

Contributions.

1. We identify **constant-memory inference** as an unnecessary restriction and show that **time-varying particle counts** are a strong inference-time scaling principle for diffusion/flow models under black-box rewards when used to front-load compute and consider a larger pool of initial noise seeds.
2. We demonstrate consistent gains of PSP over BoN, importance-sampling, and tree-search baselines at matched compute across diffusion and flow-matching backbones on **both automated metrics and human evaluation**, and show that **performance scales with compute**.
3. We show that determinism enables **offline schedule search**: pruning schedules can be simulated from cached intermediate rewards without rerunning the generator, enabling rapid per-task adaptation.

2 Related Work

2.1 Using Rewards to Steer Diffusion

A common way to use rewards for conditional generation is *gradient-based guidance* [1, 7], which backpropagates gradients of a differentiable objective to bias the sampling trajectory or the initial noise [39] towards higher reward. While effective, this requires differentiable rewards and adds significant inference overhead due to gradient computation and extra evaluations, making it less practical when rewards are black-box or expensive.

Another option is to directly *finetune* the generative backbone to maximize a reward with Reinforcement Learning [4, 46] or preference objective [41]. This can improve default sampling without requiring inference-time search, but it comes at the cost of additional training of large generative models and the associated engineering overhead. In contrast, PSP is training-free and gradient-free, making it straightforward to apply off-the-shelf across rewards and across both diffusion and flow-matching backbones.

2.2 Scaling Inference Compute with Gradient-Free Rewards

Recent methods use *intermediate* signal from black-box reward functions to scale compute at inference time efficiently by running multiple samples in parallel

allocating compute to the most promising ones. FK-Steering [34] views diffusion inference through the lens of a *Feynman–Kac interacting particle system* [5, 9] and generalizes importance-sampling approaches, like TDS [42] and SVDD [24]. It maintains a population of particles and periodically reweights and resamples them using potentials derived from intermediate reward estimates.

Another prominent line of work casts inference-time sample selection as a tree search over the denoising trajectory [23, 31, 50]. We choose DSearch [23] as their main representative in our *fixed-budget* per generation setting. It performs a beam-style search over partial trajectories, shrinking the beam over time while increasing branching (multiple children per candidate) to better estimate value from intermediate states.

Other related approaches pursue a different objective by allocating *adaptive per-prompt compute*, revisiting earlier steps for refinement and extending inference until a stopping rule indicates the generation is satisfactory [22, 50].

A shared assumption in both FK-Steering and DSearch is a *constant memory budget* during inference: the maximum number of simultaneously explored samples per step is kept roughly fixed. Our work departs from this design choice by relaxing the constant-memory constraint and instead using early pruning to spend more compute on considering a larger set of initial seeds.

2.3 Adaptive Particle Counts in Particle Filtering and Pruning

In classical particle filtering and sequential Monte Carlo, it is common to propagate a fixed number of particles at every time step, even though the complexity of the target distribution can vary significantly over time. A long-standing line of work therefore studies *adaptive* particle counts to allocate computation where it is most needed. KLD-sampling adapts the number of particles to satisfy a prescribed approximation-quality bound measured via a Kullback–Leibler criterion, using more particles in ambiguous phases and fewer once the posterior concentrates [14]. More recent “adaptive N ” particle filters adjust particle counts based on online predictive statistics and provide guarantees showing how error bounds track these updates over time [11]. In parallel, the Multi-Armed Bandit literature heavily relies on *Successive Halving* [17, 19], an algorithm that optimally distributes a fixed compute budget by evaluating a massive pool of configurations and iteratively discarding the worst-performing half. This is similar to our default pruning schedule (Sec. 4.6), but PSP is more flexible and allows for searching for optimal schedules for a given task, rewards, and prompts.

Our work can be seen as a simple application of these adaptive-particle and early-pruning ideas to reward-guided inference for diffusion image generation. Although the particle-filtering literature explores richer adaptive strategies, we show that a fixed, predetermined pruning schedule already captures much of the benefit of adaptive particle counts for conditional image generation.

The allocation principle itself is classical, yet prior reward-guided diffusion methods largely *fix parallelism* (or require stochastic branching). The closest exception is the adaptive sampler of [20], termed *rollover budget forcing*, which spends compute unevenly over the course of inference. However, they allocate

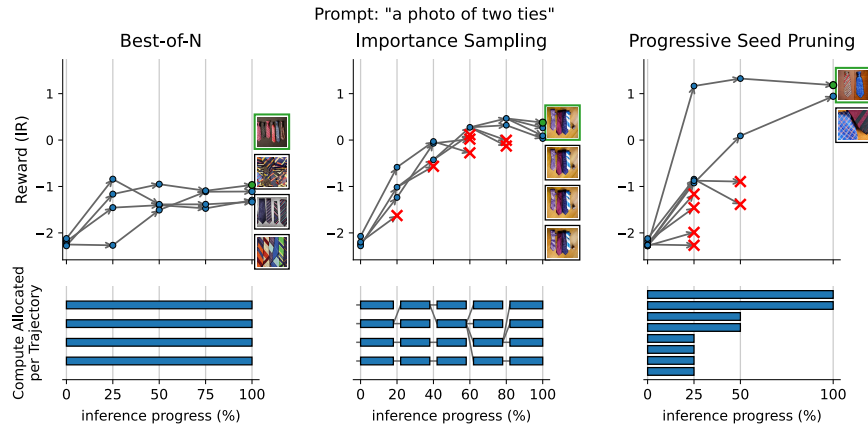


Fig. 2: Compute allocation profiles for inference-time scaling. **Top:** example reward trajectories (ImageReward) over denoising progress for multiple candidates. **Bottom:** compute allocated per candidate as a function of progress. **Left:** BoN runs N full trajectories and selects the best final reward. **Middle:** importance-sampling-style methods (e.g., FK-Steering) maintain a constant particle count and periodically resample based on intermediate rewards (red crosses indicate discarded samples). **Right:** PSP starts with more candidates and prunes low-reward trajectories early, concentrating the remaining denoising budget on a shrinking survivor set. In this illustration, PSP considers twice as many initial seeds as the other methods under the same total number of denoising steps.

in the opposite direction to us: instead of considering a larger pool of seeds early, they conserve budget early and resample more heavily later. For tasks like prompt alignment, this misplaces compute where it matters least, since the important features are coarse and thus largely fixed early in the generative process. Therefore, their adaptive sampler alone gains little over other sampling and search methods. By contrast, we show that relaxing constant memory to expand the pool of initial seeds early and prune over time is a surprisingly strong baseline for modern diffusion *and* flow-matching generators.

3 Progressive Seed Pruning

3.1 Preliminaries

Diffusion and Flow-Matching Models We consider conditional generative models that transform noise into a sample through a sequence of updates, conditioned on c (e.g., a text prompt).

Diffusion models. In diffusion [16, 36, 38], a forward noising process defines $q(x_t | x_0)$ for $t \in \{1, \dots, T\}$, and a learned model parametrizes a reverse process $p_\theta(x_{t-1} | x_t, t, c)$. In common ϵ -prediction parameterizations, the network

predicts $\epsilon_\theta(x_t, t, c)$ and a sampler (e.g., DDIM [37]) uses it to produce x_{t-1} . Importantly, the same network output yields a denoised estimate of the clean sample,

$$\hat{x}_0(x_t, t, c) \approx \mathbb{E}[x_0 \mid x_t, t, c], \quad (1)$$

which is routinely computed inside samplers and can be reused for intermediate scoring.

Flow-matching models. Flow matching models [25, 26] learn a time-dependent velocity field $v_\theta(x, t, c)$ that defines an ODE (or SDE) transporting noise to data. Sampling typically integrates the ODE from a high-noise state to a clean sample using a numerical solver (e.g., Euler Discrete [12, 13]), yielding discrete states $\{x_t\}$ at a chosen set of noise levels. With standard parameterizations used in practice, the same velocity prediction can be used to form a clean estimate via a one-step “extrapolation” to the zero-noise endpoint,

$$\hat{x}_0(x_t, t, c) = x_t - \sigma_t v_\theta(x_t, t, c), \quad (2)$$

where σ_t denotes the current noise scale in the solver schedule. As with diffusion, this estimate is already implicit in the solver update and can be extracted at negligible additional model cost.

Setup and Assumptions We assume a conditional generator like diffusion or flow matching p_θ that produces a sample by iterating discrete transitions $p_\theta(x_{t-1} \mid x_t, t, c)$ for $t \in \{T, \dots, 1\}$ from a known noise distribution. (Continuous-time solvers are handled by discretizing and applying pruning at discrete solver steps.) We also assume a black-box reward function $r(x, c)$ defined on *clean* samples (e.g., an aesthetic or preference model). Our goal is to maximize reward under a fixed inference compute budget.

Intermediate Reward Estimates PSP relies on a cheap intermediate proxy for the final sample. At each step, we compute a denoised estimate $\hat{x}_0(x_t, t, c)$ and score it:

$$s_t = r(\hat{x}_0(x_t, t, c), c). \quad (3)$$

Crucially, computing $\hat{x}_0$ reuses quantities already produced by the generator (noise prediction or velocity prediction), so PSP does not require extra generator forward passes beyond those used for denoising.

3.2 Progressive Seed Pruning (PSP)

Algorithm 1 describes PSP. The method begins from a large pool of k_T i.i.d. noise seeds and maintains a survivor set S of partial trajectories. At each step, PSP: (i) forms a denoised estimate $\hat{x}_0$ for each survivor; (ii) scores survivors using a black-box reward on $\hat{x}_0$; (iii) prunes to the top k_{t-1} survivors; and (iv) advances only survivors to the next step. By pruning early, PSP spends more of its total budget on evaluating *more* initial seeds, while still completing full denoising for the most promising trajectories.

Algorithm 1 Progressive Seed Pruning (PSP)

Input: Generator p_θ , reward $r(\cdot, c)$, condition c , steps T , schedule $\{k_t\}_{t=0}^T$ with k_T initial candidates and k_0 final survivors.
Return: Final survivors $\{x_0^i\}_{i=1}^{k_0}$
 Sample $x_T^i \sim \mathcal{N}(0, I)$ for $i \in [k_T]$
 $S \leftarrow \{x_T^i\}_{i=1}^{k_T}$
for $t \in \{T, \dots, 1\}$ **do**
 Predict clean: $\hat{x}_0^{i,t} \leftarrow \hat{x}_0(x_t^i, t, c)$ for each $x_t^i \in S$
 Score: $s_t^i \leftarrow r(\hat{x}_0^{i,t}, c)$
 Prune: $S \leftarrow \text{TopK}(S, k_{t-1})$ by scores $\{s_t^i\}$
 Denoise: sample $x_{t-1}^i \sim p_\theta(x_{t-1} \mid x_t^i, t, c)$ for each $x_t^i \in S$
 Update: $S \leftarrow \{x_{t-1}^i\}$
end for
Output: return S

Pruning intervals. In practice, we prune only at a small set of predetermined steps and keep the batch size constant between prune points. This is motivated by (a) smoothness of intermediate reward rankings across adjacent steps, and (b) systems considerations: fixed prune points with fixed survivor counts yield predictable memory usage and runtime per interval, simplifying scheduling in distributed inference.

3.3 Compute Budget and the Effective Multiplier

We measure inference compute in *denoising steps*. If an inference procedure propagates k_t concurrent trajectories at step t , the total number of generator updates is

$$C = \sum_{t=1}^T k_t. \quad (4)$$

A standard single-sample inference run has $C = T$. We therefore report the effective compute multiplier $\bar{N} = C/T$, which means that an inference procedure has the same compute as sampling $\bar{N}$ times from the regular sampler.

During inference we must both advance trajectories and compute intermediate scores. For diffusion and flow-matching models, the generator forward pass already produces the quantities needed to compute $\hat{x}_0$, so intermediate scoring does not add additional generator evaluations. Furthermore, we achieve our results with guidance from ImageReward [44], a lightweight model already used as guidance by FK-Steering [34] that adds minimal overhead to diffusion inference (see Tab. 2).

4 Experiments

4.1 Backbone Models

We evaluate on Stable Diffusion v1.5 [32] and Stable Diffusion XL [29] as diffusion models, and Stable Diffusion 3.5 (Large) [12] as a flow matching model. We use widely available implementations from HuggingFace with $T = 64$ steps for SD v1.5/SDXL and $T = 32$ solver steps for SD 3.5.

For BoN and PSP, we use deterministic solvers: DDIM [37] with $\eta = 0$ for diffusion models and Euler Discrete [12, 13] for SD 3.5. Thus, all randomness arises from the initial noise seed, matching the intended use of PSP as seed search.

FK-Steering relies on stochastic trajectories: in a deterministic setting, re-sampling generates identical children and wastes compute. For SD v1.5 and SDXL, we follow FK-Steering authors and use DDIM with $\eta = 1$ to inject noise [34]. For SD 3.5, however, strong stochasticity can noticeably degrade sampling quality under rectified-flow training, and the default stochastic setting in the public implementation yields poor FK-Steering performance. We therefore use a controlled-noise solver (Sec. S4) that injects Gaussian perturbations with a tunable scale, and we set this scale to the largest value that does not hurt baseline (non-resampled) generation quality.

4.2 Baselines and Main Comparison

Tab. 1 compares gradient-free inference-time scaling strategies under matched compute. We evaluate on GenEval prompts [15] and report: (i) the guidance reward (ImageReward [44]) of the selected output, (ii) HPSv2 [43] as an automated metric for human preference (iii) GenEval [15] as a reward-agnostic measure of prompt alignment using detector-based checks, and (iv) human evaluation of prompt-alignment. Results are averaged over 3 random seeds.

BoN runs $\bar{N}$ independent full trajectories and selects the highest-reward result (Fig. 2, left). As an importance sampling baseline, we use FK-Steering [34] (Fig. 2, middle) with the authors’ recommended hyperparameters ($\lambda = 10$, $K = 4$) and resampling schedule (20%, 40%, 60%, 80% and last step of inference progress). As a tree-search baseline, we use DSearch [23] with beam and tree width set to 2 to match the compute of other methods. We also ran experiments with other inference-time compute-scaling methods under a matched computational budget (Noise Trajectory Search (NTS) [31], Rollover Budget Forcing (RBF) [20], Breadth-First Search (BFS) [50], and SVDD [6]) but, given the cost of human studies, we report only automated metrics for these and reserve human evaluation for the main comparisons. For PSP, we use a simple fixed schedule (Fig. 2, right): start with $2\bar{N}$ seeds, prune to $\bar{N}$ at 25% progress, and prune again to $N/2$ at 50% progress.

Across all backbones, PSP improves over standard inference and outperforms all other methods at the same compute in prompt alignment, as measured by

Table 1: Main results under matched compute. We compare standard sampling ($\bar{N} = 1$), BoN, our main importance sampling (FK-Steering) and tree-search (DSearch) baselines, other relevant methods for inference-time compute scaling in diffusion (Noise Trajectory Search (NTS), Rollout Budget Forcing (RBF), Breadth-First Search (BFS), SVDD), and PSP. All results are from our experiments using the public implementation of those methods and selecting parameters to match computational budget and same reward guidance: ImageReward. We report the guidance reward (ImageReward), HPS, GenEval, and human evaluation scores on final images generated from GenEval prompts. PPS achieves the best results in prompt-alignment as measured by automated scores (GenEval) and human evaluation

Model	Sampler	T	$\bar{N}$	IR $\uparrow$	HPS $\uparrow$	GenEval $\uparrow$	Human $\uparrow$
SD v1.5	Standard	64	1	-0.159	0.257	0.434	0.431
SD v1.5	Best-of-N	64	4	0.655	0.273	0.542	0.594
SD v1.5	FK-Steering [34]	64	4	0.640	0.261	0.531	0.569
SD v1.5	DSearch [23]	64	4	0.783	0.275	0.506	0.514
SD v1.5	NTS [31]	64	4	0.485	0.272	0.478	-
SD v1.5	RBF [20]	64	4	0.670	0.274	0.520	-
SD v1.5	BFS [50]	64	4	0.820	0.263	0.564	-
SD v1.5	SVDD [6]	64	4	0.731	0.272	0.489	-
SD v1.5	PSP	64	4	0.827	0.278	0.574	0.624
SDXL	Standard	64	1	0.431	0.275	0.529	0.531
SDXL	Best-of-N	64	4	1.098	0.290	0.629	0.682
SDXL	FK-Steering [34]	64	4	1.189	0.284	0.627	0.676
SDXL	DSearch [23]	64	4	1.186	0.301	0.589	0.649
SDXL	NTS [31]	64	4	0.967	0.298	0.580	-
SDXL	RBF [20]	64	4	1.133	0.302	0.618	-
SDXL	BFS [50]	64	4	1.247	0.285	0.636	-
SDXL	SVDD [6]	64	4	0.682	0.287	0.556	-
SDXL	PSP	64	4	1.224	0.294	0.645	0.713
SD 3.5	Standard	32	1	1.045	0.297	0.713	0.787
SD 3.5	Best-of-N	32	4	1.336	0.304	0.747	0.831
SD 3.5	FK-Steering [34]	32	4	1.294	0.284	0.742	0.837
SD 3.5	DSearch [23]	32	4	1.144	0.297	0.704	0.824
SD 3.5	NTS [31]	32	4	1.086	0.295	0.699	-
SD 3.5	RBF [20]	32	4	1.253	0.296	0.738	-
SD 3.5	BFS [50]	32	4	1.343	0.285	0.747	-
SD 3.5	SVDD [6]	32	4	1.113	0.294	0.719	-
SD 3.5	PSP	32	4	1.380	0.306	0.747	0.841

GenEval score and human evaluation. Furthermore, although BFS [50] outperforms PSP in reward maximization with the SDXL backbone, PSP still outperforms BFS and all methods in GenEval and human scores. This shows that PSP is less susceptible to reward hacking [35], since it focuses on seed selection

but does not interfere during the inference process with resampling aimed at increasing reward. Supplemental experiments using HPS for reward guidance are available in Tab. S1 but IR generally leads to better generalization to GenEval, making it the best reward guidance to use in practice.

The SD 3.5 results also highlight a practical weakness of resampling methods: they require stochasticity to generate diverse children, but stochasticity can degrade flow matching generation, with BON performing better than FK-Steering. In contrast, PSP remains effective with deterministic solvers because it allocates compute to seed exploration and early elimination, not to stochastic branching.

4.3 Human Evaluation

The human evaluation results in Tab. 1 were obtained from online annotators from Prolific with prior AI-evaluation experience. They answered the question “Does the image align with the prompt?” on the images generated from GenEval prompts that were also evaluated with the automated metrics, but a single seed was used per prompt/method/backbone triplet. To our knowledge this is the first human evaluation of these methods for text-to-image. 249 annotators evaluated 8,295 images (553 per method/backbone pair, the number of GenEval prompts). Each image got 3 evaluations (24,885 ratings total) and majority vote was used to determine if an individual image aligned with its prompt. Annotator agreement was 80.3%. PSP was the best evaluated method on all three backbones.

We closely followed the protocol of [18], in which the authors use a similar study to assess differences in human and GenEval evaluations. In their study, they use only the standard samplers and the scores they obtained for them on backbones shared across studies closely match (SDXL: 0.531 vs. 0.566; SD 3.5: 0.787 vs. 0.770), reproducing their results validating the protocol.

4.4 Scaling with Compute

To test how effectively PSP uses additional compute, we scale $\bar{N}$ up to 16 by doubling the number of initial seeds and survivor counts while keeping the same fractional prune points. Fig. 3 (left) shows that PSP continues to improve with increasing $\bar{N}$ and consistently outperforms BON at matched compute.

Because PSP starts from $2\bar{N}$ deterministic trajectories, its best possible output is upper-bounded by the best fully-denoised sample among those $2\bar{N}$ seeds. Pruning introduces *regret* relative to this upper bound when the final best seed is mistakenly discarded early. Fig. 3 (middle) shows that this regret decreases as compute increases, indicating that intermediate rankings become sufficiently reliable for pruning and that PSP approaches the $2\bar{N}$ -seed upper bound using only the compute of $\bar{N}$ full trajectories.

Finally, Fig. 3 (right) illustrates that inference-time scaling can change the model-compute tradeoff: smaller backbones with larger $\bar{N}$ can outperform larger models at comparable total FLOPs, highlighting PSP as a practical knob for deployment-time quality scaling.



Fig. 3: Scaling behavior of PSP. **Left:** final guidance reward versus compute multiplier $\tilde{N}$, comparing PSP to BoNat matched compute. **Middle:** *regret* of PSP relative to BoN with double compute that would select the best seed among all $2\tilde{N}$ candidates after full denoising; regret decreases with $\tilde{N}$, indicating pruning becomes increasingly safe at larger budgets. **Right:** reward versus approximate FLOPs, showing that inference-time scaling can allow smaller models with larger $\tilde{N}$ to match or surpass larger backbones under similar compute.

4.5 Computational Overhead

PSP adds computational overhead over BoN because its intermediate samples must be decoded by the VAE and the intermediate samples must be scored by the reward model. We quantify this overhead on a single H200 in Tab. 2, comparing PSP, equivalent BoN ($N=4$), and double compute BoN ($N=8$) under the same setup as Tab. 1. Across all backbones, PSP adds under little runtime over BoN ($N=4$) and is close to $2\times$ faster than BoN ($N=8$). Relative overhead also diminishes with model size, as each diffusion step becomes proportionally more expensive. Regarding Peak VRAM usage, the built-in VAE slicing option, which decodes images sequentially through the VAE instead of in parallel. This was used in Tab. 2, adding less than 0.15 s of runtime and bringing PSP’s peak VRAM close to BoN ($N=4$) on SD v1.5 and SD 3.5. SDXL is the one case where it remains noticeably higher, because its VAE is proportionally larger. In effect, PSP ($\tilde{N}=4$) considers a BoN ($N=8$)-sized seed pool at near-BoN ($N=4$) cost.

Table 2: Computational overhead of PSP over BoN.

Model	Runtime (s)			Peak VRAM (GiB)		
	BoN ($\tilde{N}=4$)	PSP ($\tilde{N}=4$)	BoN ($\tilde{N}=8$)	BoN ($\tilde{N}=4$)	PSP ($\tilde{N}=4$)	BoN ($\tilde{N}=8$)
SD v1.5	2.65	2.99	4.70	4.29	5.30	5.31
SDXL	12.48	13.74	23.90	8.28	13.27	10.79
SD 3.5	35.97	37.59	71.71	29.85	34.85	33.86

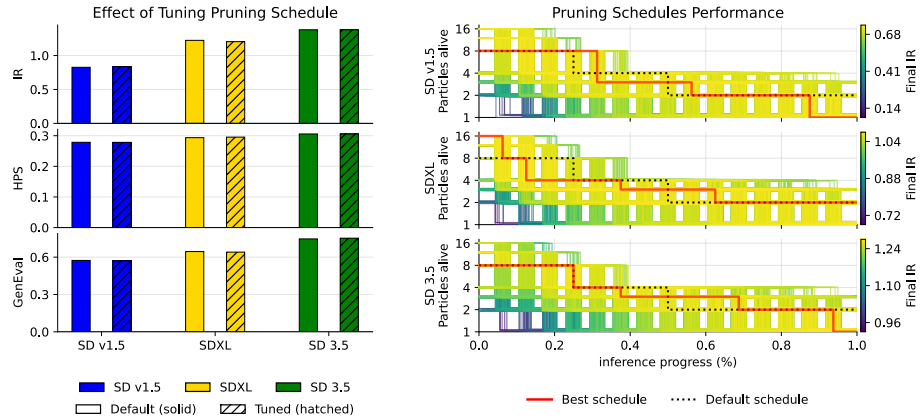


Fig. 4: Tuning the pruning strategy improves PSP. **Left.** We compare our default, off-the-shelf schedule (solid) to the best schedule found by grid search under the same compute budget (hatched). We search on prompts from the IR Benchmark and report results on prompts from GenEval, using IR as guidance signal. **Right.** Visualization of all schedules in the grid search colored by their performance on Benchmark IR. Better schedules above and jitter added for better visualization, with best and default schedules highlighted.

4.6 Tuning PSP Schedule

So far we used a simple default schedule motivated by compute matching. For large enough T , since $\sum_{i=0}^k \frac{1}{2^i} = 2 - \frac{1}{2^k}$, a geometric halving schedule lets us evaluate an arbitrarily large number of initial seeds with effective cost multiplier $\bar{N} = 2$. Scaling the schedule by a factor m (multiplying the number of particles at every stage by m) yields $\bar{N} = 2m$. Concretely, there is a tradeoff as larger k requires pruning to start earlier at less informative stages, so our default schedule (Fig. 2, right) has two pruning points, which allows considering $2\bar{N}$ initial seeds with compute multiplier $\bar{N}$ (double the seeds of BoN), and uses $m = 2$ (thus $\bar{N} = 4$), matching compute settings used in [34]. This choice makes PSP immediately usable off-the-shelf and enables a fair baseline comparison in Tab. 1 without hyperparameter tuning.

However, PSP has a rich design space: the number of initial candidates, the timing of prune points, and the survivor counts can all be varied while maintaining the same total budget. We therefore perform a grid search over schedules that preserve $\bar{N} = 4$ for each model and reward choice, select schedules that maximize reward on a held-out prompt set (Benchmark IR; details in Sec. S5), and test on prompts from GenEval. The left panel in Fig. 4 shows that tuned schedules show little to no improvement over our default schedule on guidance rewards, HPS, and GenEval scores for this task. The right panel shows all schedules searched colored by their performance on IR benchmark, highlighting the optimal chosen schedule and the default schedule we used before without tuning. We can

Table 3: PSP complements reward-based finetuning. PSP on a non-finetuned model exceeds standard sampling on a DPO-finetuned model, and applying PSP to a DPO-finetuned model yields further gains over BoN.

Model	DPO	Sampler	T	N	IR $\uparrow$	GenEval $\uparrow$
SD v1.5	✓	Standard	64	1	-0.022	0.454
SD v1.5	✓	Best-of-N	64	4	0.751	0.562
SD v1.5	✗	PSP	64	4	0.827	0.574
SD v1.5	✓	PSP	64	4	0.907	0.593
SDXL	✓	Standard	64	1	0.894	0.581
SDXL	✓	Best-of-N	64	4	1.300	0.657
SDXL	✗	PSP	64	4	1.224	0.645
SDXL	✓	PSP	64	4	1.365	0.667

see that they are not dramatically different, so our default halving strategy can be considered a good off-the-shelf configuration. Differences mostly lie in that tuned strategies tend to give up one sample close to the end in order to develop 3 samples beyond 50% of the inference process.

A key practical advantage is that when sampling is deterministic, this tuning can be done cheaply. We precompute trajectories and intermediate rewards for each prompt across a pool of initial seeds. Because the outcome of PSP is fully determined by the schedule and the precomputed trajectories, we can simulate many schedules without rerunning the generator, enabling rapid per-task tuning in domains where the default schedule might be far from the optimal one.

Fig. S2 (left) shows positive correlation between tuning (Benchmark IR) and test set (GenEval) performance across schedules, indicating that better schedules tend to generalize across prompt sets. For our default schedule, the distribution of final IR is also very similar across the two datasets (Fig. S2, middle). Fig. S2 (right) shows diminishing returns of IR on GenEval: beyond a high-IR regime, further IR gains translate weakly to GenEval, explaining the smaller GenEval improvements for strong backbones such as SD 3.5. PSP optimizes whatever guidance signal is provided, and can directly benefit from stronger reward models as they become available.

4.7 Comparison to Finetuned Models

Reward models can also be used to *finetune* generators, amortizing reward optimization across many future inference calls. We compare against DPO-finetuned backbones [41] trained on the Pick-a-Pic dataset [21]. Tab. 3 shows two take-aways. First, PSP applied to a non finetuned backbone with a modest compute multiplier outperforms standard inference on a finetuned backbone. Second, PSP and finetuning are complementary: applying PSP on a finetuned backbone yields further improvements and outperforms BoN under similar compute. Thus,

Table 4: Searching over different prompts with fixed initial seeds. PSP can optimize over any discrete set of initial conditions, including a set of prompts.

Model	Sampler	T	N	IR $\uparrow$	GenEval $\uparrow$
SD v1.5	Standard	64	1	-0.230	0.391
SD v1.5	Best-of-N	64	4	0.641	0.512
SD v1.5	PSP	64	4	0.782	0.533
SDXL	Standard	64	1	0.471	0.501
SDXL	Best-of-N	64	4	1.256	0.598
SDXL	PSP	64	4	1.319	0.604

whenever additional inference-time compute is available, PSP remains beneficial even in the presence of finetuned models.

4.8 PSP over Multiple Prompts

Although our primary focus is seed search, PSP applies to any discrete pool of candidates available at inference time. A practical example is prompt selection: modern systems often rewrite user prompts [3, 10, 12] to expand it and add detail, and different rewrites can substantially change the quality of the outcome. Quantitatively, analysis from previous work [18] has shown that prompt rewriting can bring large gains in GenEval score. We therefore apply PSP to select among multiple prompt rewrites for a *fixed* noise seed. Tab. 4 shows that PSP outperforms BoN at equal compute in this setting.

In our implementation, we generate prompt rewrites once per prompt with ChatGPT 5.2 [33] (details in Sec. S6) and treat them as the initial candidate set. This highlights another practical advantage of PSP over resampling-based methods: importance sampling over prompts would require developing a stochastic “prompt mutation” kernel that generates children prompts repeatedly during inference and also incur the additional cost of sampling from a LLM multiple times, whereas PSP only requires a fixed candidate set once per inference.

5 Discussion

We study inference-time scaling for diffusion and flow-matching models under black-box rewards. Our main contribution is showing that if we relax the constraint of constant-memory during inference, as is natural in the distributed systems that host most generative models today, then even a very simple early-pruning strategy can be more compute-efficient than stronger baselines. Across diffusion and a flow-matching backbones, PSP consistently improves reward-guided selection and achieves higher GenEval scores than BoN tree-search, and importance-sampling methods at matched compute, and continues to scale as compute increases.

A practical implication of this design is that PSP does not require stochasticity. Resampling-based approaches rely on stochastic samplers to generate diverse children from the same intermediate state. Without it, resampling collapses to repeated children and that offers no benefit over BoN. In contrast, PSP uses compute to consider more initial seeds and increasing its chances of retaining a good one. Our strong results suggest that in some tasks this is a more important and compute-efficient lever for diffusion inference than repeatedly “improving” the current best candidates via resampling. This determinism is also well aligned with the growing adoption of flow-matching over diffusion, where deterministic samplers are the standard. Moreover, because PSP is deterministic given the initial seeds, intermediate scores can be precomputed once, enabling efficient offline search over pruning schedules tailored to a specific task, reward, and prompt distribution without rerunning the generative model for each candidate schedule.

PSP has two main limitations. First, it presumes a scalar reward, so multi-objective or hard-to-quantify settings (e.g. ControlNet-style [49] constrained generation, where the signal must jointly balance text alignment with strict spatial consistency) are less natural to express as the single ranking PSP prunes on. In practice this is often surmountable: many strict-spatial tasks admit scalar perceptual losses (e.g. IoU, OKS) that PSP can rank on directly, and at matched compute it could still outperform other search/resampling methods. The deeper, shared limitation of such selection methods is the lack of *directional* guidance in space, which alternatives supply only with tradeoffs: ControlNet adds training, while gradient-based guidance requires a differentiable reward and far larger inference cost for backpropagation. Second, PSP’s advantage stems from *when* its target features are decided. For prompt alignment, the salient features are coarse and fixed early in sampling, so outcomes hinge on the initial seed and front-loading a larger seed pool pays off. For objectives dominated by fine-grained detail generated late in the trajectory, such as aesthetic quality, early seed selection matters less, and the gains shrink accordingly. This is visible in the comparatively lower HPS (which also reflects aesthetics) for SDXL in Tab. 1.

Our novelty is the finding that maintaining **variable particle counts** during inference is a strong, underexplored inference-time scaling knob for diffusion and flow-matching generators in settings where the *constant-memory* constraint is not binding, as is typical in production deployments, where inference runs on multi-GPU servers and memory is allocated elastically. Empirically, this allocation consistently beats stronger constant-memory baselines at matched generator compute and enables offline schedule tuning under deterministic sampling. Taken together, our results provide a simple but powerful design principle for inference-time scaling in modern generative models.

Acknowledgments

This study was supported by the Technology Innovation Institute (TII) through the project “Endowing AI Agents with Hierarchical Compositional & Spatial Reasoning.”

References

1. Bansal, A., Chu, H.M., Schwarzschild, A., Sengupta, R., Goldblum, M., Geiping, J., Goldstein, T.: Universal guidance for diffusion models. In: International Conference on Learning Representations. vol. 2024, pp. 51304–51323 (2024)
2. Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Giani-nazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., et al.: Graph of thoughts: Solving elaborate problems with large language models. In: Proceedings of the AAAI conference on artificial intelligence. vol. 38, pp. 17682–17690 (2024)
3. Betker, J., Goh, G., Jing, L., TimBrooks, Wang, J., Li, L., LongOuyang, JuntangZhuang, JoyceLee, YufeiGuo, WesamManassra, PrafullaDhariwal, Casey-Chu, YunxinJiao, Ramesh, A.: Improving image generation with better captions. <https://api.semanticscholar.org/CorpusID:264403242>
4. Black, K., Janner, M., Du, Y., Kostrikov, I., Levine, S.: Training diffusion models with reinforcement learning. In: International Conference on Learning Representations. vol. 2024, pp. 4965–4987 (2024)
5. Carmona, R., Fouque, J.P., Vestal, D.: Interacting particle systems for the computation of rare credit portfolio losses. *Finance and Stochastics* **13**(4), 613–633 (Sep 2009). <https://doi.org/10.1007/s00780-009-0098-8>, <http://link.springer.com/10.1007/s00780-009-0098-8>
6. Castelvocchi, D.: DeepMind’s AI helps untangle the mathematics of knots. *Nature* **600**(7888), 202–202 (Dec 2021). <https://doi.org/10.1038/d41586-021-03593-1>, <https://www.nature.com/articles/d41586-021-03593-1>
7. Chung, H., Kim, J., McCann, M.T., Klasky, M.L., Ye, J.C.: Diffusion posterior sampling for general noisy inverse problems. In: The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net (2023), <https://openreview.net/forum?id=0nD9zGAGT0k>
8. Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., Schulman, J.: Training verifiers to solve math word problems. *CoRR abs/2110.14168* (2021), <https://arxiv.org/abs/2110.14168>
9. Del Moral, P.: Feynman-Kac Formulae. Probability and its Applications, Springer New York, New York, NY (2004). <https://doi.org/10.1007/978-1-4684-9393-1>, <http://link.springer.com/10.1007/978-1-4684-9393-1>
10. Deng, C., Zhu, D., Li, K., Gou, C., Li, F., Wang, Z., Zhong, S., Yu, W., Nie, X., Song, Z., Shi, G., Fan, H.: Emerging Properties in Unified Multimodal Pretraining (Jul 2025). <https://doi.org/10.48550/arXiv.2505.14683>, <http://arxiv.org/abs/2505.14683>, arXiv:2505.14683 [cs]
11. Elvira, V., Miguez, J., Djurić, P.M.: On the performance of particle filters with adaptive number of particles. *Statistics and Computing* **31**(6), 81 (2021)
12. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: Forty-first international conference on machine learning (2024)
13. Euler, L.: Institutiones calculi integralis, vol. 1. impensis Academiae imperialis scientiarum (1792)
14. Fox, D.: KLD-sampling: Adaptive particle filters. *Advances in neural information processing systems* **14** (2001)

15. Ghosh, D., Hajishirzi, H., Schmidt, L.: Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems* **36**, 52132–52152 (2023)
16. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. *Advances in neural information processing systems* **33**, 6840–6851 (2020)
17. Jamieson, K., Talwalkar, A.: Non-stochastic best arm identification and hyperparameter optimization. In: *Artificial intelligence and statistics*. pp. 240–248. PMLR (2016)
18. Kamath, A., Chang, K.W., Krishna, R., Zettlemoyer, L., Hu, Y., Ghazvininejad, M.: GenEval 2: Addressing Benchmark Drift in Text-to-Image Evaluation (Dec 2025). <https://doi.org/10.48550/arXiv.2512.16853>, <http://arxiv.org/abs/2512.16853>, arXiv:2512.16853 [cs]
19. Karnin, Z., Koren, T., Somekh, O.: Almost optimal exploration in multi-armed bandits. In: Dasgupta, S., McAllester, D. (eds.) *Proceedings of the 30th international conference on machine learning*. *Proceedings of machine learning research*, vol. 28, pp. 1238–1246. PMLR, Atlanta, Georgia, USA (Jun 2013), <https://proceedings.mlr.press/v28/karnin13.html>, number: 3
20. Kim, J., Yoon, T., Hwang, J., Sung, M.: Inference-time scaling for flow models via stochastic generation and rollover budget forcing. *Advances in Neural Information Processing Systems* **38**, 30830–30864 (2026)
21. Kirstain, Y., Polyak, A., Singer, U., Matiana, S., Penna, J., Levy, O.: Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in neural information processing systems* **36**, 36652–36663 (2023)
22. Lee, G., Bao, T.N.N., Yoon, J., Lee, D., Kim, M., Bengio, Y., Ahn, S.: Adaptive Inference-Time Scaling via Cyclic Diffusion Search (Oct 2025). <https://doi.org/10.48550/arXiv.2505.14036>, <http://arxiv.org/abs/2505.14036>, arXiv:2505.14036 [cs]
23. Li, X., Uehara, M., Su, X., Scalia, G., Biancalani, T., Regev, A., Levine, S., Ji, S.: Dynamic search for inference-time alignment in diffusion models. *CoRR abs/2503.02039* (2025). <https://doi.org/10.48550/ARXIV.2503.02039>, <https://doi.org/10.48550/arXiv.2503.02039>
24. Li, X., Zhao, Y., Wang, C., Scalia, G., Eraslan, G., Nair, S., Biancalani, T., Ji, S., Regev, A., Levine, S., Uehara, M.: Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. In: Belgrave, D., Zhang, C., Montoya, L.N., Lin, H., Pascanu, R., Koniusz, P., Ghassemi, M., Chen, N., Ruíz, I.V.M., Loaiza-Bonilla, A. (eds.) *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025* (2025), http://papers.nips.cc/paper_files/paper/2025/hash/899af0d66d8850318a20781484416152-Abstract-Conference.html
25. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net (2023), <https://openreview.net/forum?id=PqvMRDCJT9t>
26. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net (2023), <https://openreview.net/forum?id=XVjTT1nw5z>

27. Ma, N., Tong, S., Jia, H., Hu, H., Su, Y., Zhang, M., Yang, X., Li, Y., Jaakkola, T.S., Jia, X., Xie, S.: Inference-time scaling for diffusion models beyond scaling denoising steps. CoRR **abs/2501.09732** (2025). <https://doi.org/10.48550/ARXIV.2501.09732>, <https://doi.org/10.48550/arXiv.2501.09732>
28. Oshima, Y., Suzuki, M., Matsuo, Y., Furuta, H.: Inference-time text-to-video alignment with diffusion latent beam search. *Advances in Neural Information Processing Systems* **38**, 13170–13216 (2026)
29. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: SDXL: improving latent diffusion models for high-resolution image synthesis. In: *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net (2024), <https://openreview.net/forum?id=di52zR8xgf>
30. Qi, Z., Bai, L., Xiong, H., Xie, Z.: Not all noises are created equally: diffusion noise selection and optimization. CoRR **abs/2407.14041** (2024). <https://doi.org/10.48550/ARXIV.2407.14041>, <https://doi.org/10.48550/arXiv.2407.14041>
31. Ramesh, V., Mardani, M.: Test-time scaling of diffusion models via noise trajectory search. *Advances in Neural Information Processing Systems* **38**, 87284–87317 (2026)
32. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 10684–10695 (2022)
33. Singh, A., Fry, A., Perelman, A., Tart, A., Ganesh, A., El-Kishky, A., McLaughlin, A., Low, A., Ostrow, A.J., Ananthram, A., Nathan, A., Luo, A., Helyar, A., Madry, A., Efremov, A., Spyra, A., Baker-Whitcomb, A., Beutel, A., Karpenko, A., Makelov, A., Neitz, A., Wei, A., Barr, A., Kirchmeyer, A., Ivanov, A., Christakis, A., Gillespie, A., Tam, A., Bennett, A., Wan, A., Huang, A., Sandjideh, A.M., Yang, A., Kumar, A., Saraiva, A., Vallone, A., Gheorghe, A., Garcia, A.G., Braunstein, A., Liu, A., Schmidt, A., Mereskin, A., Mishchenko, A., Applebaum, A., Rogerson, A., Rajan, A., Wei, A., Kotha, A., Srivastava, A., Agrawal, A., Vijayvergiya, A., Tyra, A., Nair, A., Nayak, A., Eggers, B., Ji, B., Hoover, B., Chen, B., Chen, B., Barak, B., Minaiev, B., Hao, B., Baker, B., Lightcap, B., McKinzie, B., Wang, B., Quinn, B., Fioca, B., Hsu, B., Yang, B., Yu, B., Zhang, B., Brenner, B., Zetino, C.R., Raymond, C., Lugaresi, C., Paz, C., Hudson, C., Whitney, C., Li, C., Chen, C., Cole, C., Voss, C., Ding, C., Shen, C., Huang, C., Colby, C., Hallacy, C., Koch, C., Lu, C., Kaplan, C., Kim, C., Minott-Henriques, C.J., Frey, C., Yu, C., Czarnecki, C., Reid, C., Wei, C., Decareaux, C., Scheau, C., Zhang, C., Forbes, C., Tang, D., Goldberg, D., Roberts, D., Palmie, D., Kappler, D., Levine, D., Wright, D., Leo, D., Lin, D., Robinson, D., Grabb, D., Chen, D., Lim, D., Salama, D., Bhattacharjee, D., Tsipras, D., Li, D., Yu, D., Strouse, D.J., Williams, D., Hunn, D., Bayes, E., Arbus, E., Akyurek, E., Le, E.Y., Widmann, E., Yani, E., Proehl, E., Sert, E., Cheung, E., Schwartz, E., Han, E., Jiang, E., Mitchell, E., Sigler, E., Wallace, E., Ritter, E., Kavanaugh, E., Mays, E., Nikishin, E., Li, F., Such, F.P., Peres, F.d.A.B., Raso, F., Bekerman, F., Tsimpouras, F., Chantzis, F., Song, F., Zhang, F., Rila, G., McGrath, G., Briggs, G., Yang, G., Parascandolo, G., Chabot, G., Kim, G., Zhao, G., Valiant, G., Leclerc, G., Salman, H., Wang, H., Sheng, H., Jiang, H., Wang, H., Jin, H., Sikchi, H., Schmidt, H., Aspegren, H., Chen, H., Qiu, H., Lightman, H., Covert, I., Kivlichan, I., Silber, I., Sohl, I., Hammoud, I., Clavera, I., Lan, I., Akkaya, I., Kostrikov, I., Kofman, I., Etinger, I., Singal, I., Hehir, J., Huh, J., Pan, J., Wilczynski, J., Pachocki, J., Lee, J., Quinn, J., Kiros, J., Kalra, J., Samaroo, J., Wang, J., Wolfe, J., Chen, J., Wang, J., Harb,

- J., Han, J., Wang, J., Zhao, J., Chen, J., Yang, J., Tworek, J., Chand, J., Landon, J., Liang, J., Lin, J., Liu, J., Wang, J., Tang, J., Yin, J., Jang, J., Morris, J., Flynn, J., Ferstad, J., Heidecke, J., Fishbein, J., Hallman, J., Grant, J., Chien, J., Gordon, J., Park, J., Liss, J., Kraaijeveld, J., Guay, J., Mo, J., Lawson, J., McGrath, J., Vendrow, J., Jiao, J., Lee, J., Steele, J., Wang, J., Mao, J., Chen, K., Hayashi, K., Xiao, K., Salahi, K., Wu, K., Sekhri, K., Sharma, K., Singhal, K., Li, K., Nguyen, K., Gu-Lemberg, K., King, K., Liu, K., Stone, K., Yu, K., Ying, K., Georgiev, K., Lim, K., Tirumala, K., Miller, K., Ahmad, L., Lv, L., Clare, L., Fauconnet, L., Itow, L., Yang, L., Romaniuk, L., Anise, L., Byron, L., Pathak, L., Maksin, L., Lo, L., Ho, L., Jing, L., Wu, L., Xiong, L., Mamitsuka, L., Yang, L., McCallum, L., Held, L., Bourgeois, L., Engstrom, L., Kuhn, L., Feuvrier, L., Zhang, L., Switzer, L., Kondraciuk, L., Kaiser, L., Joglekar, M., Singh, M., Shah, M., Stratta, M., Williams, M., Chen, M., Sun, M., Cayton, M., Li, M., Zhang, M., Aljube, M., Nichols, M., Haines, M., Schwarzer, M., Gupta, M., Shah, M., Huang, M., Dong, M., Wang, M., Glaese, M., Carroll, M., Lampe, M., Malek, M., Sharman, M., Zhang, M., Wang, M., Pokrass, M., Florian, M., Pavlov, M., Wang, M., Chen, M., Wang, M., Feng, M., Bavarian, M., Lin, M., Abdool, M., Rohaninejad, M., Soto, N., Staudacher, N., LaFontaine, N., Marwell, N., Liu, N., Preston, N., Turley, N., Ansman, N., Blades, N., Pancha, N., Mikhaylin, N., Felix, N., Handa, N., Rai, N., Keskar, N., Brown, N., Nachum, O., Boiko, O., Murk, O., Watkins, O., Gleeson, O., Mishkin, P., Lesiewicz, P., Baltescu, P., Belov, P., Zhokhov, P., Pronin, P., Guo, P., Thacker, P., Liu, Q., Yuan, Q., Liu, Q., Dias, R., Puckett, R., Arora, R., Mullapudi, R.T., Gaon, R., Miyara, R., Song, R., Aggarwal, R., Marsan, R.J., Yemiru, R., Xiong, R., Kshirsagar, R., Nuttall, R., Tsiupa, R., Eldan, R., Wang, R., James, R., Ziv, R., Shu, R., Nigmatullin, R., Jain, S., Talaie, S., Altman, S., Arnesen, S., Toizer, S., Toyer, S., Miserendino, S., Agarwal, S., Yoo, S., Heon, S., Ethersmith, S., Grove, S., Taylor, S., Bubeck, S., Banerjee, S., Amdo, S., Zhao, S., Wu, S., Santurkar, S., Zhao, S., Chaudhuri, S.R., Krishnaswamy, S., Shuaiqi, Xia, Cheng, S., Anadkat, S., Fishman, S.P., Tobin, S., Fu, S., Jain, S., Mei, S., Egoian, S., Kim, S., Golden, S., Mah, S.Q., Lin, S., Imm, S., Sharpe, S., Yadlowsky, S., Choudhry, S., Eum, S., Sanjeev, S., Khan, T., Stramer, T., Wang, T., Xin, T., Gogineni, T., Christianson, T., Sanders, T., Patwardhan, T., Degry, T., Shadwell, T., Fu, T., Gao, T., Garipov, T., Sriskandarajah, T., Sherbakov, T., Kaftan, T., Hiratsuka, T., Wang, T., Song, T., Zhao, T., Peterson, T., Kharitonov, V., Chernova, V., Kosaraju, V., Kuo, V., Pong, V., Verma, V., Petrov, V., Jiang, W., Zhang, W., Zhou, W., Xie, W., Zhan, W., McCabe, W., DePue, W., Ellsworth, W., Bain, W., Thompson, W., Chen, X., Qi, X., Xiang, X., Shi, X., Dubois, Y., Yu, Y., Khakbaz, Y., Wu, Y., Qian, Y., Lee, Y.T., Chen, Y., Zhang, Y., Xiong, Y., Tian, Y., Cha, Y., Bai, Y., Yang, Y., Yuan, Y., Li, Y., Zhang, Y., Yang, Y., Jin, Y., Jiang, Y., Wang, Y., Wang, Y., Liu, Y., Stubenvoll, Z., Dou, Z., Wu, Z., Wang, Z.: OpenAI GPT-5 System Card (Dec 2025). <https://doi.org/10.48550/arXiv.2601.03267>, <http://arxiv.org/abs/2601.03267>, arXiv:2601.03267 [cs]
34. Singhal, R., Horvitz, Z., Teehan, R., Ren, M., Yu, Z., Mckeown, K., Ranganath, R.: A general framework for inference-time scaling and steering of diffusion models. In: Singh, A., Fazel, M., Hsu, D., Lacoste-Julien, S., Berkenkamp, F., Maharaj, T., Wagstaff, K., Zhu, J. (eds.) Proceedings of the 42nd international conference on machine learning. Proceedings of machine learning research, vol. 267, pp. 55810–55827. PMLR (Jul 2025), <https://proceedings.mlr.press/v267/singhal25b.html>

35. Skalse, J., Howe, N., Krashennnikov, D., Krueger, D.: Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems* **35**, 9460–9471 (2022)
36. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: *International conference on machine learning*. pp. 2256–2265. pmlr (2015)
37. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3–7, 2021*. OpenReview.net (2021), <https://openreview.net/forum?id=St1giarCHLP>
38. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456* (2020)
39. Tang, Z., Peng, J., Tang, J., Hong, M., Wang, F., Chang, T.: Inference-time alignment of diffusion models with direct noise optimization. In: Singh, A., Fazel, M., Hsu, D., Lacoste-Julien, S., Berkenkamp, F., Maharaj, T., Wagstaff, K., Zhu, J. (eds.) *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13–19, 2025*. *Proceedings of Machine Learning Research*, vol. 267. PMLR / OpenReview.net (2025), <https://proceedings.mlr.press/v267/tang25h.html>
40. Valmeekam, K., Marquez, M., Sreedharan, S., Kambhampati, S.: On the planning abilities of large language models—a critical investigation. *Advances in neural information processing systems* **36**, 75993–76005 (2023)
41. Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., Naik, N.: Diffusion model alignment using direct preference optimization. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 8228–8238 (2024)
42. Wu, L., Trippe, B., Naesseth, C., Blei, D., Cunningham, J.P.: Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems* **36**, 31372–31403 (2023)
43. Wu, X., Hao, Y., Sun, K., Chen, Y., Zhu, F., Zhao, R., Li, H.: Human Preference Score v2: A Solid Benchmark for Evaluating Human Preferences of Text-to-Image Synthesis (Sep 2023). <https://doi.org/10.48550/arXiv.2306.09341>, <http://arxiv.org/abs/2306.09341>, arXiv:2306.09341 [cs]
44. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems* **36**, 15903–15935 (2023)
45. Xu, K., Zhang, L., Shi, J.: Good seed makes a good crop: Discovering secret seeds in text-to-image diffusion models. In: *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. pp. 3024–3034. IEEE (2025)
46. Yang, K., Tao, J., Lyu, J., Ge, C., Chen, J., Shen, W., Zhu, X., Li, X.: Using human feedback to fine-tune diffusion models without any reward model. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 8941–8951 (2024)
47. Yao, S., Chen, H., Yang, J., Narasimhan, K.: Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems* **35**, 20744–20757 (2022)
48. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., Narasimhan, K.: Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems* **36**, 11809–11822 (2023)

- 49. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3836–3847 (2023)
- 50. Zhang, X., Lin, H., Ye, H., Zou, J., Ma, J., Liang, Y., Du, Y.: Inference-time Scaling of Diffusion Models through Classical Search (Oct 2025). <https://doi.org/10.48550/arXiv.2505.23614>, <http://arxiv.org/abs/2505.23614>, arXiv:2505.23614 [cs]