References

[1] Beatrice Achilli, Enrico Ventura, Gianluigi Silvestri, Bao Pham, Gabriel Raya, Dmitry Krotov, Carlo Lucibello, and Luca Ambrogioni. Losing dimensions: Geometric memorization in generative diffusion. *arXiv preprint arXiv:2410.08727*, 2024. 8

[2] Donghoon Ahn, Hyoungwon Cho, Jaewon Min, Wooseok Jang, Jungwoo Kim, SeonHwa Kim, Hyun Hee Park, Kyong Hwan Jin, and Seungryong Kim. Self-rectifying diffusion sampling with perturbed-attention guidance. *arXiv preprint arXiv:2403.17377*, 2024. 7, 2

[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. 2, 3

[4] Hyungjin Chung, Jeongsol Kim, Geon Yeong Park, Hyelin Nam, and Jong Chul Ye. Cfg++: Manifold-constrained classifier free guidance for diffusion models. *arXiv preprint arXiv:2406.08070*, 2024. 7, 2

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. 1, 2, 6

[7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 6

[8] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 2

[9] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016. 2

[10] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 6

[11] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 1, 2

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 1

[13] Susung Hong, Gyuseong Lee, Wooseok Jang, and Seungryong Kim. Improving sample quality of diffusion models using self-attention guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7462–7471, 2023. 7, 1

[14] Yi-Ting Hsiao, Siavash Khodadadeh, Kevin Duarte, Wei-An Lin, Hui Qu, Mingi Kwon, and Ratheesh Kalarot. Plug-and-play diffusion distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13743–13752, 2024. 8

[15] Black Forest Labs. FLUX. https://github.com/black-forest-labs/flux, 2024. Accessed: 2024-11-15. 8

[16] John M Lee. *Introduction to Riemannian manifolds*. Springer, 2018. 3

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 6

[18] Peter Lorenz, Ricard L Durall, and Janis Keuper. Detecting images generated by deep diffusion models using their local intrinsic dimensionality. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 448–459, 2023. 8

[19] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 1

[20] Kazusato Oko, Shunta Akiyama, and Taiji Suzuki. Diffusion models are minimax optimal distribution estimators. In *International Conference on Machine Learning*, pages 26517–26582. PMLR, 2023. 3

[21] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 2, 6

[22] Jakiw Pidstrigach. Score-based generative models detect manifolds. *Advances in Neural Information Processing Systems*, 35:35852–35865, 2022. 3

[23] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 2, 6

[24] Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021. 3

[25] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 6

[26] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 2

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 1, 6

[28] Brendan Leigh Ross, Hamidreza Kamkari, Tongzi Wu, Rasa Hosseinzadeh, Zhaoyan Liu, George Stein, Jesse C Cresswell, and Gabriel Loaiza-Ganem. A geometric framework for understanding memorization in generative models. *arXiv preprint arXiv:2411.00113*, 2024. 7

[29] Seyedmorteza Sadat, Manuel Kansy, Otmar Hilliges, and Romann M Weber. No training, no problem: Rethinking classifier-free guidance for diffusion models. *arXiv preprint arXiv:2407.02687*, 2024. 7

[30] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 1

[31] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 1

[32] Jan Pawel Stanczuk, Georgios Batzolis, Teo Deveney, and Carola-Bibiane Schönlieb. Diffusion models encode the intrinsic dimension of data manifolds. In *Forty-first International Conference on Machine Learning*, 2024. 2, 3, 4

[33] Zhenyue Zhang and Hongyuan Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM journal on scientific computing*, 26(1):313–338, 2004. 3

TCFG: Tangential Damping Classifier-free Guidance

Supplementary Material

9. Computational Efficiency of SVD in Our Method

Our method requires performing SVD with only two components: the unconditional score and the conditional score. As a result, the computational time required for this operation is negligible. Tab. 4 illustrates the additional time introduced by the SVD calculation.

The computational cost varies depending on the image resolution, as higher resolutions require larger dimensional SVD computations. For instance, the time required for SVD in SDv3 with a 1024 resolution is greater than that for SDv1.5 with a 256 resolution. However, even in the case of SDv3, the time taken remains under 0.1 seconds per image, accounting for less than a 0.01

For memory usage, even with SD v3 (the largest latent dimensions), the additional memory was only 18.48 MB. In Figures 2, 3, and 4 of Section Intuition, we highlight that SVD requires only two tensors. Our design choice (full_matrices=False during SVD) further optimizes memory, resulting in memory complexity: $\text{Memory}_{\text{reduced}} \approx O(m+n)$. Since $n = 2$, memory usage scales linearly with the latent dimension m.

10. Toy Example Experiment Setup

In the toy example experiment, we utilized the `two moons` dataset from scikit-learn. The two moons were conditioned on labels 0 and 1, while label 2 was used for the unconditional setting. The setup followed the standard DDPM configuration with 100 timesteps for training. The noise schedule employed a linear beta schedule with $\beta_{\text{min}} = 0.0001$ and $\beta_{\text{max}} = 0.02$. The network consisted of two linear layers, trained using the Adam optimizer with a learning rate of 0.001 for 5,000 iterations.

11. Verifying Cosine Similarity Across Singular Vectors

Fig. 3 demonstrates that the cosine similarity between the singular vectors of the unconditional and conditional scores is significantly high for indices close to 0. However, the order of indices may differ between the unconditional and conditional scores. To ensure that the results in Figure 3 are not influenced by differing index orders, we conducted the experiment shown in Fig. 11.

In this experiment, we measured the cosine similarity of all 17,000 singular vectors based on the text conditional score, ensuring that each singular vector was used only once by selecting and plotting the highest similarity value for each singular vector without duplication. The results consistently show that high similarity is observed only for lower indices, corroborating the findings of the original experiment. This confirms that the observed pattern is not due to the order of indices but rather reflects the fact that singular vectors corresponding to high singular values are indeed similar.

	NFE	Execution Time (s)	Time Difference (s)	Percentage Difference (%)
SD v1.5	50	2.556	-	-
SD v1.5 + ours		2.577	0.021	+ 0.008
SDXL	50	13.176	-	-
SDXL + ours		13.221	0.045	+ 0.003
SD v3	40	19.473	-	-
SD v3 + ours		19.558	0.085	+ 0.004

Table 4. Comparison of execution times for standard diffusion models and our method across different resolutions and models. The additional time introduced by our method is negligible, with percentage differences remaining below 0.01% in all cases.

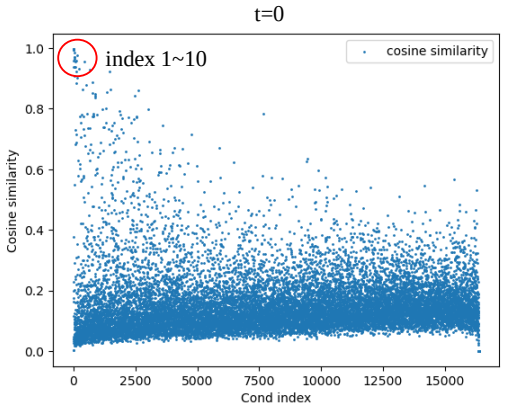

Figure 11. Cosine similarity between singular vectors of unconditional and conditional scores. We measured the cosine similarity of all 17,000 singular vectors based on the text conditional score order, ensuring that each singular vector was used only once by selecting and plotting the highest similarity value for each singular vector without duplication.

12. Compatibility of Our Method with Other Techniques

Our method modifies unconditional scores using text conditions, making it compatible with other approaches. For instance, in SAG [13], the unconditional score is derived by blurring the attention map. We applied our projection

	FID	CLIPScore
SDXL turbo	21.47	0.31
SDXL turbo + ours	**20.36**	**0.32**
InstaFlow	16.76	**0.30**
InstaFlow + ours	**16.19**	**0.30**
PixArt-Σ	22.53	**0.32**
PixArt-Σ + ours	**20.19**	**0.32**

Table 5. Performance comparison of our method applied to SDXL Turbo, InstaFlow, and PixArt-Σ. FID scores decrease while CLIPScore remains the same or improves, confirming the broad applicability of our method across different generation models, including high-resolution models.

Model	Scheduler	CFG scale	Sampling steps	etc
SD v1.4	PNDMScheduler	7.5	50	SAG scale: 0.75
SD v1.5	PNDMScheduler	7.5	50	PAG scale: 3.0
SDXL	EulerDiscreteScheduler	5.0	50	CFG++ scale: 0.6
SD v3	FlowMatchEulerDiscreteScheduler	7.0	28	
SDXL Turbo	EulerAncestralDiscreteScheduler	2.0	1	
InstaFlow	PNDMScheduler	7.5	1	
PixArt-Σ	DPMSolverMultistepScheduler	4.5	20	

Table 6. Experimental details.

method to the unconditional score used in SAG, and Fig. 12 demonstrates improved results when combined with our method.

In PAG [2], an additional score is used alongside CFG, where the self-attention map is set to identity. We observed that projecting the perturbed-attention guidance score in PAG did not yield significant improvements, likely because this score differs fundamentally from the CFG unconditional score. Instead, we projected the unconditional score used in PAG's CFG computation using TCFG, resulting in enhanced image details and structure. Please refer to Fig. 12.

CFG++ [4] proposes an interpolation-based CFG computation method instead of extrapolation. When we applied our projection to the unconditional score used in CFG++, as shown in Fig. 13, the results improved further. These findings highlight the versatility of our method and its ability to enhance other existing techniques.

13. Experimental details.

We provide details on the sampler, guidance scale, sampling steps, and additional existing baselines' hyperparameters in Tab. 6.

14. Additional Results: Few-Step and High-Resolution Image Generation

We further report the application of our method to few-step generation models and high-resolution image generation. Tab. 5 presents the results when our method is applied to SDXL Turbo (a one-step generation model) and InstaFlow (also a one-step generation model). In both cases, FID scores improve, while CLIPScore remains the same or improves, demonstrating that our method performs effectively not only in many-step models but also across all models utilizing CFG. Notably, for SDXL Turbo, the CFG scale was set to a very low value of 1.3.

Additionally, Tab. 5 highlights the performance of our method in PixArt-Σ, a high-resolution text-to-image generation model. Similar improvements are observed, with a reduction in FID scores and maintenance of CLIPScore. Fig. 14 showcases the visual results of PixArt-Σ, further validating the effectiveness of our approach.

```python
if self.do_classifier_free_guidance:
    noise_pred_uncond, noise_pred_text = noise_pred.chunk(2)

    all_noise = torch.stack((noise_pred_text, noise_pred_uncond), dim=1).to(dtype=torch.float32)
    all_noise = all_noise.reshape(all_noise.size(0), all_noise.size(1), -1)

    U, S, Vh = torch.linalg.svd(all_noise, full_matrices=False)
    Vh = Vh.to(all_noise.device)
    Vh_modified = Vh.clone().to(all_noise.device)
    Vh_modified[:,1] = 0
    noise_null_flat = noise_pred_uncond.reshape(noise_pred_uncond.size(0), 1, -1).to(dtype=torch.float32)
    noise_null_flat = noise_null_flat.to(Vh.device)
    x_Vh = torch.matmul(noise_null_flat, Vh.transpose(-2, -1))
    x_Vh_V = torch.matmul(x_Vh, Vh_modified)
    noise_pred_uncond = x_Vh_V.reshape(*noise_pred_uncond.shape).to(noise_pred_text.dtype).to(noise_pred_text.device)
    noise_pred = noise_pred_uncond + self.guidance_scale * (noise_pred_text - noise_pred_uncond)
```

Listing 1. Code for TCFG with the Hugging Face code style.

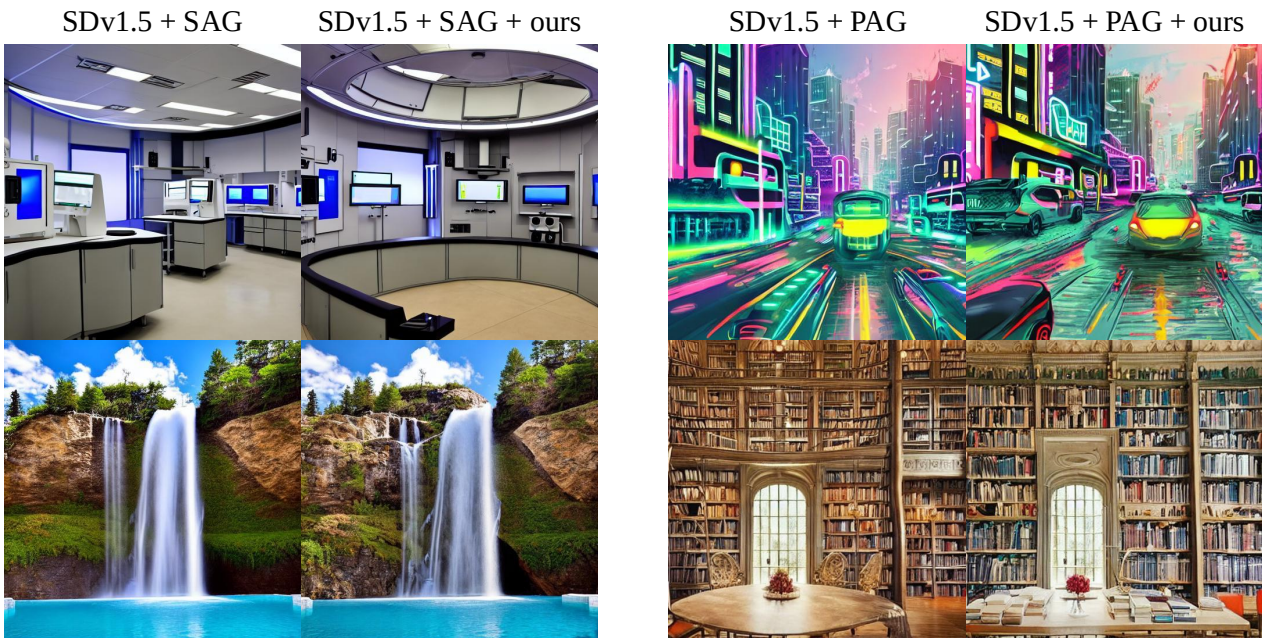

Figure 12. We observed that incorporating our method with SAG and PAG approaches improved the image structure, details, and overall color quality.

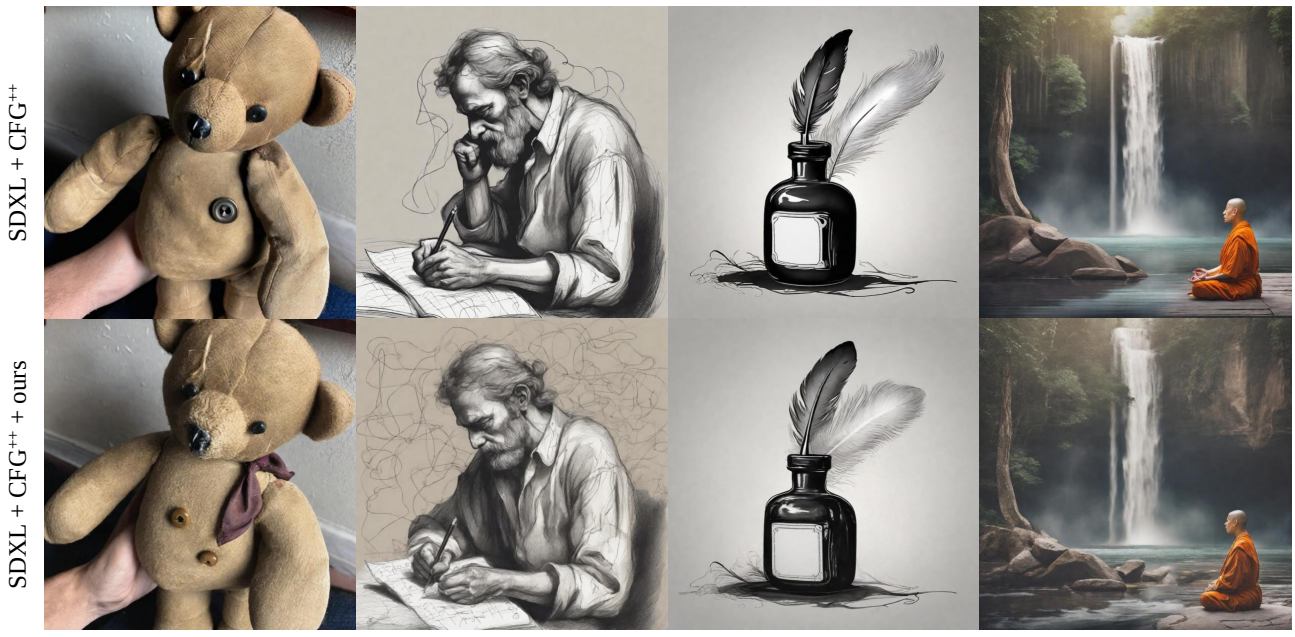

Figure 13. We observed that incorporating our method with CFG++ approaches improved the image structure, details, and overall color quality.

Figure 14. Visual examples generated by PixArt-Σ with our method, demonstrating improved image quality in terms of structure, details, and overall aesthetics