NoiseCtrl: A Sampling-Algorithm-Agnostic Conditional Generation Method for Diffusion Models

Supplementary Material

Proof for Fast Estimation in Polar Coordinates

Theorem:
Let $\boldsymbol{\psi}'$ be a randomly chosen point in $\mathbb{C}$. By using $\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}) \approx \mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')+\left[\nabla_{\boldsymbol{\psi}'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')\right]^T(\boldsymbol{\psi}-\boldsymbol{\psi}')$, optimization (**??**) simplifies to:

$$\begin{aligned}&\arg\min_{\boldsymbol{\psi}} \left[\nabla_{\boldsymbol{\psi}'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')\right]^T(\boldsymbol{\psi}-\boldsymbol{\psi}')\\ &\text{s.t. } \boldsymbol{\psi}\in\mathbb{C}\end{aligned} \tag{1}$$

This linear optimization achieves its minima at the boundary of $\mathbb{C}$. The analytical expression thus is:

$$\psi_k^* = \begin{cases} 0, & \text{if } \nabla_{\psi_k'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') \geq 0 \\ \pi-\epsilon, & \text{if } \nabla_{\psi_k'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') < 0 \end{cases} \tag{2}$$

$$\phi^* = \begin{cases} 0, & \text{if } \nabla_{\phi'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') \geq 0 \\ 2\pi-\epsilon, & \text{if } \nabla_{\phi'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') < 0 \end{cases} \tag{3}$$

where ϵ is a very small positive number used to ensure that the variables approach their upper bounds without exceeding them.

Proof: The theorem approximates $\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi})$ using the expression $\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')+\left[\nabla_{\boldsymbol{\psi}'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')\right]^T(\boldsymbol{\psi}-\boldsymbol{\psi}')$. Since $\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')$ is a constant, the optimization problem (**??**) simplifies to:

$$\begin{aligned}&\arg\min_{\boldsymbol{\psi}} \left[\nabla_{\boldsymbol{\psi}'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')\right]^T(\boldsymbol{\psi}-\boldsymbol{\psi}')\\ &\text{s.t. } \boldsymbol{\psi}\in\mathbb{C}\end{aligned} \tag{4}$$

Given that $\boldsymbol{\psi}'$ is a constant, this optimization can be further reduced to:

$$\begin{aligned}&\arg\min_{\boldsymbol{\psi}} \left[\nabla_{\boldsymbol{\psi}'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')\right]^T\boldsymbol{\psi}\\ &\text{s.t. } \boldsymbol{\psi}\in\mathbb{C}\end{aligned} \tag{5}$$

Since $\left[\nabla_{\boldsymbol{\psi}'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}')\right]^T\boldsymbol{\psi}$ is a linear function, its minimum is achieved at the boundary of $\mathbb{C}$. Therefore, we have:

$$\psi_k^* = \begin{cases} 0, & \text{if } \nabla_{\psi_k'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') \geq 0 \\ \pi-\epsilon, & \text{if } \nabla_{\psi_k'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') < 0 \end{cases} \tag{6}$$

$$\phi^* = \begin{cases} 0, & \text{if } \nabla_{\phi'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') \geq 0 \\ 2\pi-\epsilon, & \text{if } \nabla_{\phi'}\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}') < 0 \end{cases} \tag{7}$$

Proof for Fast Estimation in Cartesian Coordinates

Theorem:
Substituting the angle vector $\boldsymbol{\psi}$ in polar coordinates with a direction vector $\mathbf{d}$ in cartesian coordinates offers an alternative method for determining the direction. To be precise, let $\mathbb{S}$ represent the sphere surface with radius 1, the linear optimization problem (**??**) can be transformed to

$$\begin{aligned}&\arg\min_{\mathbf{d}} \left[\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')\right]^T (\|\mathbf{z}'\|_2\mathbf{d} - \mathbf{z}') \\ &\text{s.t. } \mathbf{d} \in \mathbb{S}\end{aligned} \tag{8}$$

where $\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}') = \mathcal{E}(\hat{z}_{0|t}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_{t+1}, t+1) + \boldsymbol{\sigma}_{t+1}\mathbf{z}'), \boldsymbol{y})$ and $\mathbf{z}'$ is a random Gaussian point. According to [1], the closed-form solution of this manifold optimization problem can be formulated as

$$\mathbf{d} = -\frac{\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')}{||\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')||_2}. \tag{9}$$

Proof: According to Equation (**??**), we have

$$\mathcal{L}_{\boldsymbol{y}}(\boldsymbol{\psi}) = \mathcal{E}(\hat{z}_{0|t}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_{t+1}, t+1) + \boldsymbol{\sigma}_{t+1}\boldsymbol{T}(\mathrm{r}_{t+1}, \boldsymbol{\psi})), \boldsymbol{y}) \tag{10}$$

Due to randomness of r_{t+1}, it is reasonable to take a Gaussian sample $\mathbf{z}$ to substitute $\boldsymbol{\psi}$, and we thus have:

$$\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}) = \mathcal{E}(\hat{z}_{0|t}(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_{t+1}, t+1) + \boldsymbol{\sigma}_{t+1}\boldsymbol{z}), \boldsymbol{y}) \tag{11}$$

This equation implies that we take the direction on the sphere surface with radius $\|\mathbf{z}\|_2$ to represent the angular $\boldsymbol{\psi}$. Thus the optimization problem (4) can be reduced to

$$\begin{aligned}&\arg\min_{\mathbf{z}} \left[\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')\right]^T (\mathbf{z} - \mathbf{z}') \\ &\text{s.t. } \mathbf{z} \in \|\mathbf{z}'\|_2\mathbb{S}\end{aligned} \tag{12}$$

where $\mathbf{z}'$ is a random Gaussian point and $\|\mathbf{z}'\|_2\mathbb{S}$ denotes the sphere surface with the radius $\|\mathbf{z}'\|_2$. We can further transform it to

$$\begin{aligned}&\arg\min_{\mathbf{d}} \left[\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')\right]^T (\|\mathbf{z}'\|_2\mathbf{d} - \mathbf{z}') \\ &\text{s.t. } \mathbf{d} \in \mathbb{S}\end{aligned} \tag{13}$$

Due to the constant $\mathbf{z}'$, we have

$$\begin{aligned}&\arg\min_{\mathbf{d}} \left[\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')\right]^T \mathbf{d} \\ &\text{s.t. } \mathbf{d} \in \mathbb{S}\end{aligned} \tag{14}$$

According to [1], the closed-form solution of this manifold optimization problem can be formulated as

$$\mathbf{d} = -\frac{\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')}{||\nabla_{\mathbf{z}'}\mathcal{L}'_{\boldsymbol{y}}(\mathbf{z}')||_2}. \tag{15}$$

Proof for κ Investigation

Theorem:

$$\mathbb{E}[\mathbf{d}^T\mathbf{u}] = \frac{\kappa}{\kappa + (n-1)}. \tag{16}$$

Proof: For a Von Mises–Fisher distribution with mean direction $\mathbf{u}$ and concentration κ, the expected value is

$$\mathbb{E}[\mathbf{u}] = \frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}(\kappa)}\mathbf{u} \tag{17}$$

We want to compute the expected value of the dot product between the random vector $\mathbf{u}$ and the mean direction $\mathbf{d}$.

$$\mathbb{E}[\mathbf{d}^\top \mathbf{u}] = \int_{S^{n-1}} \mathbf{d}^T \boldsymbol{u} \cdot \frac{\kappa^{\frac{n}{2}-1}}{(2\pi)^{\frac{n}{2}} I_{\frac{n}{2}-1}(\kappa)} \exp(\kappa \mathbf{d}^T \boldsymbol{u})\, d\mathbf{d}.$$

The von Mises-Fisher distribution is isotropic, meaning that it is symmetric with respect to rotations around the direction of the mean vector $\mathbf{d}$. Thus, without loss of generality, we can assume that $\mathbf{d} = (0, 0, \cdots, 1)$ by rotating the coordinate system. This simplification reduces the problem to computing the first moment of the distribution where the mean direction is aligned with the last coordinate. Thus, we have

$$\mathbb{E}[\mathbf{d}^T \mathbf{u}] = \frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}(\kappa)} \tag{18}$$

We observe that SciPy offers the 'scipy.special' module to compute the modified Bessel function, specifically $\mathcal{I}_{\frac{n}{2}-1}(\kappa)$. However, this implementation is not effective for large values of κ. Consequently, it becomes infeasible to directly compute the ratio $\frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}}(\kappa)$. To address this issue, we propose an approximation method. Notably, the modified Bessel function is defined as follows:

$$I_\nu(\kappa) = \sum_{m=0}^{\infty} \frac{1}{m!\Gamma(m+\nu+1)} \left(\frac{\kappa}{2}\right)^{2m+\nu}$$

The recurrence relation for modified Bessel functions is given by:

$$I_{\nu-1}(\kappa) = \frac{2(\nu-1)}{\kappa} I_\nu(\kappa) + I'_\nu(\kappa)$$

and similarly:

$$I_{\nu+1}(\kappa) = \frac{2\nu}{\kappa} I_\nu(\kappa) - I'_\nu(\kappa)$$

For $\nu = \frac{n}{2}$, we can specifically write:

$$I_{\frac{n}{2}-1}(\kappa) = \frac{n-2}{\kappa} I_{\frac{n}{2}}(\kappa) + I'_{\frac{n}{2}}(\kappa)$$

The ratio of modified Bessel functions $\frac{I_{\frac{n}{2}-1}(\kappa)}{I_{\frac{n}{2}}(\kappa)}$ can be computed by substituting the recurrence relation into the expression for the ratio. Starting with the recurrence relation:

$$\frac{I_{\frac{n}{2}-1}(\kappa)}{I_{\frac{n}{2}}(\kappa)} = \frac{\frac{n-2}{\kappa} I_{\frac{n}{2}}(\kappa) + I'_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}}(\kappa)}$$

Dividing through by $I_{\frac{n}{2}}(\kappa)$, we get:

$$\frac{I_{\frac{n}{2}-1}(\kappa)}{I_{\frac{n}{2}}(\kappa)} = \frac{n-2}{\kappa} + \frac{I'_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}}(\kappa)}$$

For large κ, $\frac{I'_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}}(\kappa)} \approx 1$. Substituting this into the ratio:

$$\frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}(\kappa)} \approx \frac{\kappa}{\kappa + (n-2)}$$

For small values of κ, particularly as $\kappa \to 0$, the modified Bessel functions $I_\nu(\kappa)$ have the following asymptotic behavior:
1. $I_{\frac{n}{2}}(\kappa) \sim \frac{\kappa^{\frac{n}{2}}}{2^{\frac{n}{2}}\Gamma\left(\frac{n}{2}\right)}$ for small κ
2. $I_{\frac{n}{2}-1}(\kappa) \sim \frac{\kappa^{\frac{n}{2}-1}}{2^{\frac{n}{2}-1}\Gamma\left(\frac{n}{2}-1\right)}$ for small κ

As $\kappa \to 0$, we can look at the ratio of these two Bessel functions:

$$\frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}(\kappa)} \sim \frac{\frac{\kappa^{\frac{n}{2}}}{2^{\frac{n}{2}}\Gamma\left(\frac{n}{2}\right)}}{\frac{\kappa^{\frac{n}{2}-1}}{2^{\frac{n}{2}-1}\Gamma\left(\frac{n}{2}-1\right)}} = \frac{\kappa}{2} \cdot \frac{\Gamma\left(\frac{n}{2}-1\right)}{\Gamma\left(\frac{n}{2}\right)}$$

At $\kappa = 0$, the ratio approaches 0 because κ is the dominant factor in the numerator, and the ratio is proportional to κ. Therefore, we have:

$$\frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}(\kappa)} \to 0 \quad \text{as} \quad \kappa \to 0$$

This behavior is analogous to the simplified expression $\frac{\kappa}{\kappa+(n-1)}$.

Combining the insights from the previous discussion, we find that for both large and small values of κ, the ratio of the modified Bessel functions of the first kind can be approximated as follows:

$$\frac{I_{\frac{n}{2}}(\kappa)}{I_{\frac{n}{2}-1}(\kappa)} \approx \frac{\kappa}{\kappa + (n-1)} \approx \frac{\kappa}{\kappa + (n-2)}$$

This approximation provides a useful simplification for computational purposes, especially when dealing with the ratio of Bessel functions. It highlights the behavior of these functions in different regimes of κ, offering a practical tool for numerical computations and theoretical analyses.

DDPM$_{\text{fast},2}$: An Example of NoiseCtrl

Algorithm 1 DDPM$_{\text{fast},2}$ Sampling

```
1: **Input:** pure noise $\mathbf{z}_T$, iteration number $T$, concentration parameter $\kappa$.
2: **for** $t = T$ **to** $1$ **do**
3:     $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_t, t) = \frac{1}{\sqrt{\alpha_t}}\mathbf{z}_t - \frac{1-\alpha_t}{\sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{z}_t, t)$
4:     Determine $\mathbf{d}$ by solving optimization (**??**)
5:     $\boldsymbol{\epsilon}'_t = \mathrm{r}\mathbf{u}$, where $\mathrm{r} \sim f_{\mathrm{r}}(\mathrm{r}),\ \mathbf{u} \sim f_{\mathbf{u};\mathbf{d},\kappa}(\mathbf{u}; \mathbf{d}, \kappa)$
6:     $\mathbf{z}_{t-1} = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{z}_t, t) + \sigma_t \boldsymbol{\epsilon}'_t$
7: **end for**
8: **Return** $\mathbf{z}_0$
```

```
```
noise_scheduler = DDPMScheduler.from_pretrained(path, subfolder="scheduler")
for t in timesteps:
    # Algorithm 1 Line3: Default reverse process
    latent_model_input = torch.cat([latents] * 2)
    noise_pred = unet(latent_model_input, t, text_embeddings).sample
    noise_pred_uncond, noise_pred_text = noise_pred.chunk(2)
    noise_pred = noise_pred_uncond + 7.5 * (noise_pred_text - noise_pred_uncond)
    prev_latent = noise_scheduler.step(latents, t, noise_pred)

    # Algorithm 1 Line4: Compute gradient to obtain the direction
    torch.enable_grad(True)
    variance_noise = torch.randn_like(prev_latent)
    x = variance_noise.detach().clone().requires_grad_(True)
    prev_x = prev_sample + sigma_t * x
    prev_t = noise_scheduler.previous_timestep(t)
    pred_cond = cond_net(prev_x, prev_t)
    loss = torch.nn.functional.mse_loss(
        pred_cond.float(), target.float(), reduction="mean")
    grad = torch.autograd.grad(
        loss, x, grad_outputs=torch.ones_like(loss), create_graph=True)[0]
    grad_direction = grad / torch.norm(grad.view(x.shape[0], -1), dim=-1)

    # Algorithm 1 Line5: Compute conditional noise
    kappa = torch.Tensor([1e+4]).cuda()
    grad_direction = VonMisesFisher(
        grad_direction.flatten(), kappa).rsample(1)[0].reshape(1, c, h, w)
    radius=chi2_dist.sample((bsz,)).cuda().sqrt()
    variance_noise = (-radius * grad_direction)
    variance_noise = variance_noise.detach().requires_grad_(False)
    torch.enable_grad(False)

    # Algorithm 1 Line6: compute the final previous timestep latents
    latents = prev_sample + sigma_t * variance_noise

```
```

More Results of NoiseCtrl with Fast Direction Estimation and the Second-Type Distribution

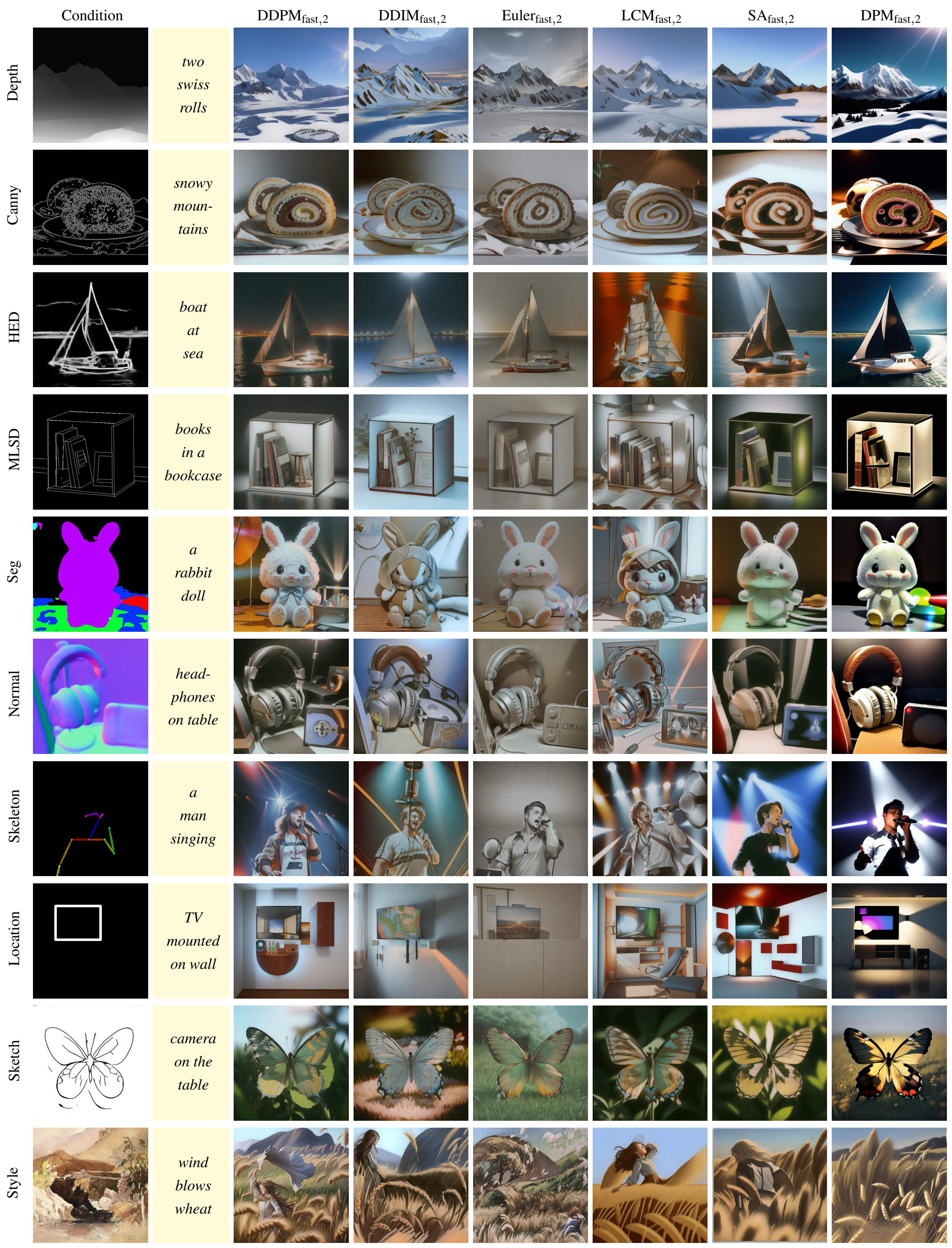

Figure 1. Visual Illustration for NoiseCtrl with Fast Direction Estimation and the First-Type Distribution as described in Equation (**??**).

Visual Comparison For NoiseCtrl with Four Configurations and Six Sampling Algorithms

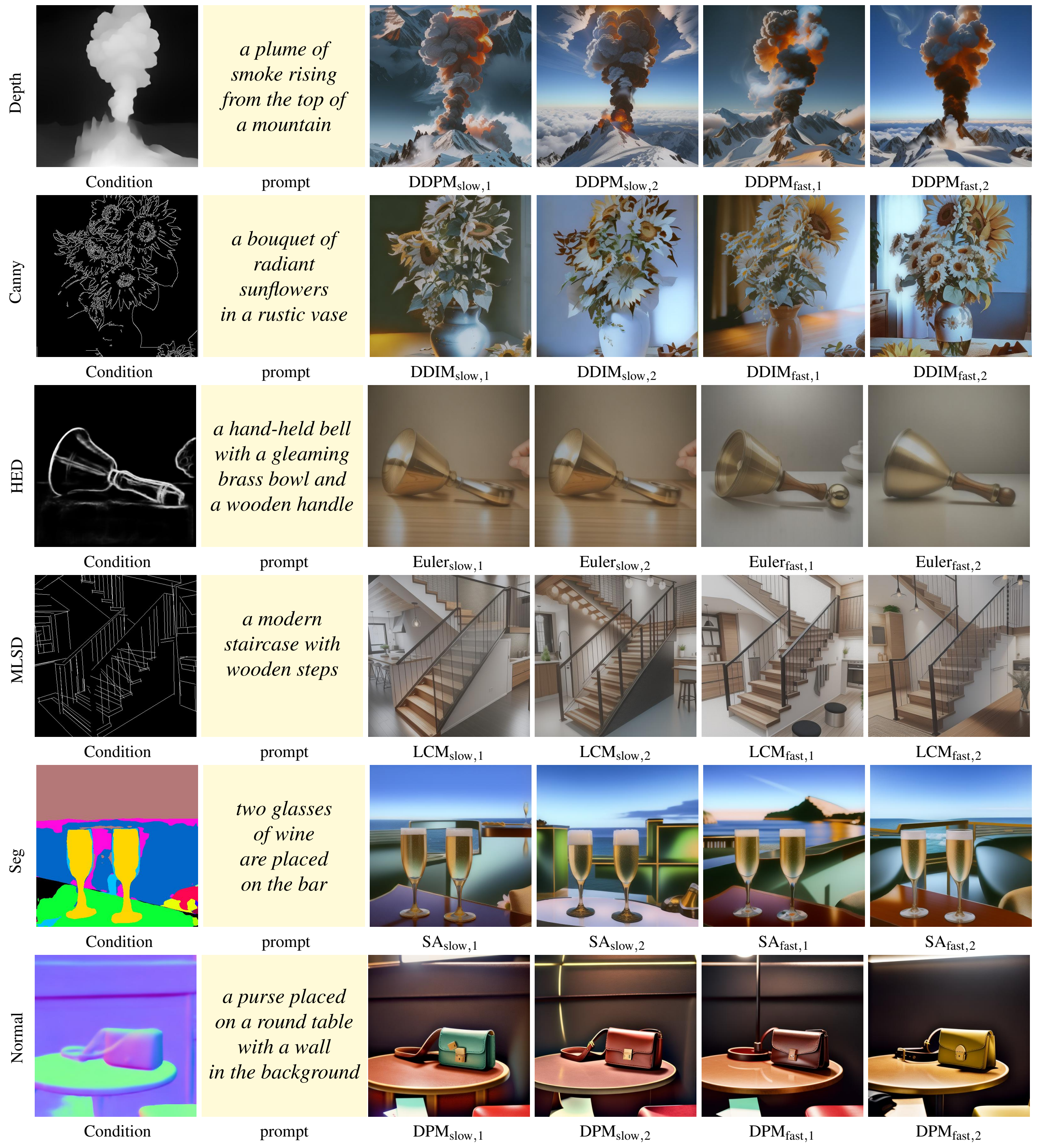

Figure 2. NoiseCtrl Visual Comparison with Four Configurations (fast/slow direction estimation methods and two noise sampling distributions (**??**) (**??**)) and Six Sampling Algorithms (DDPM, DDIM, Euler, LCM, SA and DPM) for Six Conditions (Depth, Canny, HED, M-LSD, Seg and Normal). The first column delineates the conditions, the second column records the text prompts. Columns three and four document the performance of four distinct NoiseCtrl configurations, with each row corresponding to a different sampling algorithm. It is straightforward to confirm that all four NoiseCtrl variants are capable of yielding satisfactory results.

References

[1] Lingxiao Yang, Shutong Ding, Yifan Cai, Jingyi Yu, Jingya Wang, and Ye Shi. Guidance with Spherical Gaussian Constraint for Conditional Diffusion. In *International Conference on Machine Learning*, 2024. 2