Conditional Simulation Using Diffusion Schrödinger Bridges (Supplementary Material)

Yuyang Shi[1] **Valentin De Bortoli**[2] **George Deligiannidis**[1] **Arnaud Doucet**[1]

[1]Department of Statistics, University of Oxford, UK
[2]ENS, PSL University, Paris, France

A ORGANIZATION OF THE SUPPLEMENTARY

The supplementary is organized as follows. We recall the DSB algorithm for unconditional simulation from De Bortoli et al. [2021] in Appendix B. The proofs of our propositions are given in Appendix C. In Appendix D, we give details on the loss functions we use to train CDSB. A continuous-time version of the conditional time-reversal and conditional DSB is presented in Appendix E. The forward-backward technique used in our experiments is detailed in Appendix F. Finally, we provide experimental details and guidelines in Appendix G.

B DIFFUSION SCHRÖDINGER BRIDGE

We recall here the DSB algorithm introduced by De Bortoli et al. [2021] which is a numerical approximation of IPF[1].

Algorithm 2 Diffusion Schrödinger Bridge [De Bortoli et al., 2021]

for $n \in \{0, \dots, L\}$ **do**
  while not converged **do**
    Sample $\{X_k^j\}_{k,j=0}^{N,M}$, where $X_0^j \sim p_{\text{data}}$, and $X_{k+1}^j = \mathbf{F}_{\phi^n}(k, X_k^j) + \sqrt{2\gamma_{k+1}} Z_{k+1}^j$
    Compute $\hat{\ell}_n^b(\theta^n)$ approximating (8)
    $\theta^n \leftarrow$ Gradient Step$(\hat{\ell}_n^b(\theta^n))$
  end while
  while not converged **do**
    Sample $\{X_k^j\}_{k,j=0}^{N,M}$, where $X_N^j \sim p_{\text{ref}}$, and $X_{k-1}^j = \mathbf{B}_{\theta^n}(k, X_k^j) + \sqrt{2\gamma_k} \tilde{Z}_k^j$
    Compute $\hat{\ell}_{n+1}^f(\phi^{n+1})$ approximating (9)
    $\phi^{n+1} \leftarrow$ Gradient Step$(\hat{\ell}_{n+1}^f(\phi^{n+1}))$
  end while
end for
Output: (θ^L, ϕ^{L+1})

In this (unconditional) SB scenario, the transition kernels satisfy $q_{k|k+1}^n(x|x') = \mathcal{N}(x; \mathbf{B}_{\theta^n}(k+1, x'), 2\gamma_{k+1}\,\mathrm{Id})$ and $p_{k+1|k}^n(x'|x) = \mathcal{N}(x'; \mathbf{F}_{\phi^n}(k, x), 2\gamma_{k+1}\,\mathrm{Id})$ where θ^n is obtained by minimizing

$$\ell_n^b(\theta) = \mathbb{E}_{p^n}[\textstyle\sum_k \|\mathbf{B}_\theta(k+1, X_{k+1}) - G_{n,k}(X_k, X_{k+1})\|^2] \tag{8}$$

[1]For discrete measures, IPF is also known as the Sinkhorn algorithm and can be implemented exactly [Peyré and Cuturi, 2019].

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

for $G_{n,k}(x, x') = x' + \mathbf{F}_{\phi^n}(k, x) - \mathbf{F}_{\phi^n}(k, x')$ and ϕ^{n+1} by minimizing

$$\ell^f_{n+1}(\phi) = \mathbb{E}_{q^n}[\sum_k \|\mathbf{F}_\phi(k, X_k) - H_{n,k}(X_k, X_{k+1})\|^2] \tag{9}$$

for $H_{n,k}(x, x') = x + \mathbf{B}_{\theta^n}(k+1, x') - \mathbf{B}_{\theta^n}(k+1, x)$. See De Bortoli et al. [2021] for a derivation of these loss functions.

C PROOFS OF PROPOSITIONS

C.1 PROOF OF PROPOSITION 1

Let $\bar{\pi}$ such that $\mathrm{KL}(\bar{\pi}|\bar{p}) < +\infty$, which exists since we have that $\mathrm{KL}(\bar{\pi}^\star|\bar{p}) < +\infty$, and $\bar{\pi}_0 = p_{\text{join}}$, $\bar{\pi}_N = p_{\text{jref}}$, where we define the joint forward process $\bar{p}(x_{0:N}, y_{0:N}) := p_{y_0}(x_{0:N})\bar{p}_{\text{obs}}(y_{0:N})$. Recall that $p_{y_0}(x_{0:n}) := p(x_0|y_0)\prod_{k=0}^{N-1} p_{k+1|k}(x_{k+1}|x_k)$ is the forward process starting from the posterior $p(x_0|y_0)$, and $\bar{p}_{\text{obs}}(y_{0:N}) := p_{\text{obs}}(y_0)\prod_{k=0}^{N-1}\delta_{y_k}(y_{k+1})$ is the extended y-process. Since $\mathrm{KL}(\bar{\pi}|\bar{p}) < +\infty$ we have using the transfer theorem [Kullback, 1997, Theorem 2.4.1] that $\mathrm{KL}(\bar{\pi}_{\text{obs}}|\bar{p}_{\text{obs}}) < +\infty$, where $\bar{\pi}_{\text{obs}}(y_{0:N}) := \int_{(\mathbb{R}^d)^N} \bar{\pi}(x_{0:N}, y_{0:N})\mathrm{d}x_{0:N}$. In addition, using the chain rule for the Kullback–Leibler divergence, see [Léonard, 2014, Theorem 2.4], we get that

$$\mathrm{KL}(\bar{\pi}_{\text{obs}}|\bar{p}_{\text{obs}}) = \mathrm{KL}(\bar{\pi}_{\text{obs},0}|p_{\text{obs}}) + \int_{\mathcal{Y}} \mathrm{KL}(\bar{\pi}_{\text{obs}|0}|\bar{p}_{\text{obs}|0})p_{\text{obs}}(y)\mathrm{d}y < +\infty,$$

where $\bar{p}_{\text{obs}|0} = \prod_{k=0}^{N-1}\delta_{y_k}(y_{k+1})$ and therefore $\bar{\pi}_{\text{obs}|0} = \bar{p}_{\text{obs}|0}$. Since we also have that $\bar{\pi}_{\text{obs},0} = p_{\text{obs}}$ we get that $\bar{\pi}_{\text{obs}} = \bar{p}_{\text{obs}}$. Hence, letting π^c be the kernel such that $\bar{\pi} = \pi^c \otimes \bar{p}_{\text{obs}}$ we have using [Léonard, 2014, Theorem 2.4] that

$$\mathrm{KL}(\bar{\pi}|\bar{p}) = \int_{\mathcal{Y}} \mathrm{KL}\left(\pi^c_y|p_y\right) p_{\text{obs}}(y)\mathrm{d}y. \tag{10}$$

In addition, we have $\bar{\pi}_0 = \pi^c_0 \otimes p_{\text{obs}} = p_{\text{join}}$. Similarly, we have $\bar{\pi}_N = \pi^c_N \otimes p_{\text{obs}} = p_{\text{jref}}$. Hence, $\pi^c_{y,0} = p(\cdot|y)$ and $\pi^c_{y,N} = p_{\text{ref}}$, p_{obs}-almost surely. Let $\bar{\pi}^\star = \pi^{\star,c} \otimes \bar{p}_{\text{obs}}$ be the minimizer of (5) and $\hat{\pi}^c$ be the minimizer of (4). Then, we have that $\bar{\pi} = \hat{\pi}^c \otimes \bar{p}_{\text{obs}}$ satisfies $\mathrm{KL}(\bar{\pi}^\star|\bar{p}) \leq \mathrm{KL}(\bar{\pi}|\bar{p})$. Using (10), we have that $\mathbb{E}[\mathrm{KL}\left(\pi^{\star,c}_Y|p_Y\right)] \leq \mathbb{E}[\mathrm{KL}\left(\hat{\pi}^c_Y|p_Y\right)]$. But we have that $\mathbb{E}[\mathrm{KL}\left(\hat{\pi}^c_Y|p_Y\right)] \leq \mathbb{E}[\mathrm{KL}\left(\pi^{\star,c}_Y|p_Y\right)]$ since $\hat{\pi}^c$ is the minimizer of (4). Using the uniqueness of the minimizer of (4) we have that $\pi^{\star,c} = \hat{\pi}^c$, which concludes the proof.

C.2 PROOF OF PROPOSITION 2

Let $n \in \mathbb{N}$ and $\bar{q}$ be such that $\mathrm{KL}(\bar{q}|\bar{p}^n) < +\infty$ and $\bar{q}_N = p_{\text{jref}}$ (note that the existence of such a distribution is ensured since $\mathrm{KL}(p_{\text{join}} \otimes p_{\text{jref}}|\bar{p}^n_{0,N}) < +\infty$). Using the chain rule for the Kullback–Leibler divergence, see [Léonard, 2014, Theorem 2], we have

$$\mathrm{KL}(\bar{q}|\bar{p}^n) = \mathrm{KL}\left(\bar{q}_{\text{obs}}|\bar{p}_{\text{obs}}\right) + \int_{\mathcal{Y}^{N+1}} \mathrm{KL}(\bar{q}_{|\text{obs}}|\bar{p}^n_{|\text{obs}})\mathrm{d}\bar{q}_{\text{obs}}(y_{0:N}), \tag{11}$$

where $\bar{q}_{\text{obs}} = \int_{\mathcal{X}^{N+1}} \bar{q}(x_{0:N}, y_{0:N})\mathrm{d}x_{0:N}$ and $\bar{q}_{|\text{obs}}$ and $\bar{p}^n_{|\text{obs}}$ are the conditional distribution of $\bar{q}$, respectively $\bar{p}^n$ w.r.t. to $y_{0:N}$. Since $\mathrm{KL}\left(\bar{q}_{\text{obs}}|\bar{p}_{\text{obs}}\right) < +\infty$, we can use [Léonard, 2014, Theorem 2.4] and we have

$$\mathrm{KL}\left(\bar{q}_{\text{obs}}|\bar{p}_{\text{obs}}\right) = \mathrm{KL}\left(\bar{q}_{\text{obs},N}|\bar{p}_{\text{obs},N}\right) + \int_{\mathcal{Y}} \mathrm{KL}\left(\bar{q}_{\text{obs}|N}|\bar{p}_{\text{obs}|N}\right) \mathrm{d}\bar{q}_{\text{obs},N}(y_N),$$

with $\bar{p}_{\text{obs}|N}(y_{0:N-1}|y_N) = \prod_{k=0}^{N-1}\delta_{y_{k+1}}(y_k)$. Therefore, since $\mathrm{KL}\left(\bar{q}_{\text{obs}}|\bar{p}_{\text{obs}}\right) < +\infty$, we get that $\bar{q}_{\text{obs}|N}(y_{0:N-1}|y_N) = \prod_{k=0}^{N-1}\delta_{y_{k+1}}(y_k)$. Since $\bar{q}_{\text{obs},N} = p_{\text{obs}}$, we get that $\bar{q}(x_{0:N}, y_{0:N}) = \bar{p}_{\text{obs}}(y_{0:N})\bar{q}(x_{0:N}|y_{0:N}) = \bar{p}_{\text{obs}}(y_{0:N})\bar{q}(x_{0:N}|y_N)$, where we have used that $y_N = y_k$ for $k \in \{0, \ldots, N\}$, $\bar{p}_{\text{obs}}(y_{0:N})$ almost surely. Combining this result and (11) we get that

$$\mathrm{KL}(\bar{q}|\bar{p}^n) = \int_{\mathcal{Y}^{N+1}} \mathrm{KL}(\bar{q}_{|\text{obs}}|\bar{p}^n_{|\text{obs}})\mathrm{d}p_{\text{obs}}(y_{0:N}) = \int_{\mathcal{Y}} \mathrm{KL}(\bar{q}(\cdot|y_N)|\bar{p}^n(\cdot|y_N))\mathrm{d}p_{\text{obs}}(y_N),$$

Using [Léonard, 2014, Theorem 2], we have that for any $y_N \in \mathcal{Y}$

$$\mathrm{KL}(\bar{q}(\cdot|y_N)|\bar{p}^n(\cdot|y_N) = \mathrm{KL}(p_{\text{ref}}|\bar{p}^n_N(\cdot|y_N)) + \int_{\mathcal{Y}} \mathrm{KL}(\bar{q}(\cdot|y_N, x_N)|\bar{p}^n(\cdot|y_N, x_N))p_{\text{ref}}(x_N)\mathrm{d}x_N.$$

For the IPF solution $\bar{q}^n$, we get that $\bar{q}^n(\cdot|y_N, x_N) = \bar{p}^n(\cdot|y_N, x_N)$. Therefore for any $x_{0:N} \in \mathcal{X}^{N+1}$ and $y_N \in \mathcal{Y}$,

$$\bar{q}^n(x_{0:N}|y_N) = p_{\text{ref}}(x_N)\prod_{k=0}^{N-1} \bar{p}^n_{k|k+1}(x_k|x_{k+1}, y_N).$$

The proof is similar for any $x_{0:N} \in \mathcal{X}^{N+1}$ and $y_0 \in \mathcal{Y}$, we have

$$\bar{p}^{n+1}(x_{0:N}|y_0) = p(x_0|y_0)\prod_{k=0}^{N-1} \bar{q}^n_{k+1|k}(x_{k+1}|x_k, y_0).$$

C.3 PROOF OF PROPOSITION 3

Using [Léger, 2021, Corollary 1], we get that for any $n \in \mathbb{N}$ with $n \geq 1$

$$\mathrm{KL}(\bar{\pi}_0^n | p_{\text{join}}) + \mathrm{KL}(\bar{\pi}_N^n | p_{\text{jref}}) \leq \frac{2}{n} \mathrm{KL}(\bar{\pi}^\star | \bar{p}). \tag{12}$$

Similarly to Proposition 2, we have that for any $n \in \mathbb{N}$, there exists a Markov kernel $\pi^{c,n}$ such that $\bar{\pi}^n = \bar{p}_{\text{obs}} \otimes \pi^{c,n}$. Recall that there exists a Markov kernel $\pi^{c,\star}$ such that $\bar{\pi}^\star = \bar{p}_{\text{obs}} \otimes \pi^{c,\star}$ and that $\bar{p} = \bar{p}_{\text{obs}} \otimes p_y$. Hence, using [Léonard, 2014, Theorem 2.4], we get that for any $n \in \mathbb{N}$,

$$\mathrm{KL}(\bar{\pi}_0^n | p_{\text{join}}) = \mathbb{E}[\mathrm{KL}(\pi_{Y,0}^{c,n} | p(\cdot|Y))], \qquad \mathrm{KL}(\bar{\pi}_N^n | p_{\text{jref}}) = \mathbb{E}[\mathrm{KL}(\pi_{Y,N}^{c,n} | p_{\text{ref}})]. \tag{13}$$

Similarly, we have that

$$\mathrm{KL}(\bar{\pi}^\star | \bar{p}) = \mathbb{E}[\mathrm{KL}(\pi_Y^{c,\star} | p_Y)]. \tag{14}$$

We conclude the proof upon combining (12), (13) and (14).

D DETAILS ON THE LOSS FUNCTIONS

In this section, we simplify notation and write Y for all the random variables $Y_0, Y_1, ..., Y_N$ as they are all equal almost surely under $\bar{p}^n$ and $\bar{q}^n$, similarly to Section 4. In Section 4, the transitions satisfy $\bar{q}^n_{k|k+1}(x|x', y) = \mathcal{N}(x; \mathbf{B}^y_{\theta^n}(k+1, x'), 2\gamma_{k+1}\,\mathrm{Id})$ and $\bar{p}^n_{k+1|k}(x'|x, y) = \mathcal{N}(x'; \mathbf{F}^y_{\phi^n}(k, x), 2\gamma_{k+1}\,\mathrm{Id})$ where θ^n is obtained by minimizing

$$\ell_n^b(\theta) = \mathbb{E}_{\bar{p}^n}[\textstyle\sum_k \|\mathbf{B}^Y_\theta(k+1, X_{k+1}) - G^Y_{n,k}(X_k, X_{k+1})\|^2]$$

for $G^y_{n,k}(x, x') = x' + \mathbf{F}^y_{\phi^n}(k, x) - \mathbf{F}^y_{\phi^n}(k, x')$ and ϕ^{n+1} by minimizing

$$\ell_{n+1}^f(\phi) = \mathbb{E}_{\bar{q}^n}[\textstyle\sum_k \|\mathbf{F}^Y_\phi(k, X_k) - H^Y_{n,k}(X_k, X_{k+1})\|^2]$$

for $H^y_{n,k}(x, x') = x + \mathbf{B}^y_{\theta^n}(k+1, x') - \mathbf{B}^y_{\theta^n}(k+1, x)$. We justify these formulas by proving the following result which is a straightforward extension of De Bortoli et al. [2021]. We recall that for any $n \in \mathbb{N}$, $k \in \{0, \ldots, N\}$, $x_k, x_{k+1} \in \mathbb{R}^d$ and $y \in \mathcal{Y}$, $b^{n,y}_{k+1}(x_{k+1}) = -f^{n,y}_k(x_{k+1}) + 2\nabla \log \bar{p}^n_{k+1}(x_{k+1}|y)$ and $f^{n+1,y}_k(x_k) = -b^{n,y}_{k+1}(x_k) + 2\nabla \log \bar{q}^n_k(x_k|y)$.[2]

Proposition 4. *Assume that for any $n \in \mathbb{N}$ and $k \in \{0, \ldots, N-1\}$, $\bar{q}_k(\cdot|y)$ and $\bar{p}_k(\cdot|y)$ are bounded and*

$$\bar{q}^n_{k|k+1}(x_k|x_{k+1}, y) = \mathcal{N}(x_k; B^{n,y}_{k+1}(x_{k+1}), 2\gamma_{k+1}\,\mathrm{Id}),\ \bar{p}^n_{k+1|k}(x_{k+1}|x_k, y) = \mathcal{N}(x_{k+1}; F^{n,y}_k(x_k), 2\gamma_{k+1}\,\mathrm{Id}),$$

with $B^{n,y}_{k+1}(x) = x + \gamma_{k+1} b^{n,y}_{k+1}(x)$, $F^{n,y}_k(x) = x + \gamma_{k+1} f^{n,y}_k(x)$ for any $x \in \mathbb{R}^d$. Then we have for any $n \in \mathbb{N}$ and $k \in \{0, \ldots, N-1\}$

$$B^n_{k+1} = \arg\min_{\mathrm{B} \in \mathrm{L}^2(\mathbb{R}^d \times \mathcal{Y}, \mathbb{R}^d)} \mathbb{E}_{\bar{p}^n}[\|\mathrm{B}(X_{k+1}, Y) - G^Y_{n,k}(X_k, X_{k+1})\|^2], \tag{15}$$

$$F^{n+1}_k = \arg\min_{\mathrm{F} \in \mathrm{L}^2(\mathbb{R}^d \times \mathcal{Y}, \mathbb{R}^d)} \mathbb{E}_{\bar{q}^n}[\|\mathrm{F}(X_k, Y) - H^Y_{n,k}(X_k, X_{k+1})\|^2], \tag{16}$$

$$G^y_{n,k}(x, x') = x' + F^{n,y}_k(x) - F^{n,y}_k(x'), \qquad H^y_{n,k}(x, x') = x + B^{n,y}_{k+1}(x') - B^{n,y}_{k+1}(x).$$

Proof. We only prove (15) since the proof (16) is similar. Let $n \in \mathbb{N}$ and $k \in \{0, \ldots, N-1\}$. For any $x_{k+1} \in \mathbb{R}^d$ we have

$$\bar{p}^n_{k+1}(x_{k+1}|y) = (4\pi\gamma_{k+1})^{-d/2} \textstyle\int_{\mathbb{R}^d} \bar{p}^n(x_k|y) \exp[-\|F^{n,y}_k(x_k) - x_{k+1}\|^2/(4\gamma_{k+1})] \mathrm{d}x_k,$$

with $F^{n,y}_k(x_k) = x_k + \gamma_{k+1} f^{n,y}_k(x_k)$. Since $\bar{p}^n_k > 0$ is bounded using the dominated convergence theorem we have for any $x_{k+1} \in \mathbb{R}^d$

$$\nabla_{x_{k+1}} \log \bar{p}^n_{k+1}(x_{k+1}|y) = \textstyle\int_{\mathbb{R}^d} (F^{n,y}_k(x_k) - x_{k+1})/(2\gamma_{k+1})\, \bar{p}_{k|k+1}(x_k|x_{k+1}, y) \mathrm{d}x_k.$$

Therefore we get that for any $x_{k+1} \in \mathbb{R}^d$

$$b^{n,y}_{k+1}(x_{k+1}) = \textstyle\int_{\mathbb{R}^d} (F^{n,y}_k(x_k) - F^{n,y}_k(x_{k+1}))/\gamma_{k+1}\, \bar{p}_{k|k+1}(x_k|x_{k+1}, y) \mathrm{d}x_k.$$

[2] We should have conditioned w.r.t. y_N and y_0 but since $y_0 = y_1 = \cdots = y_N$ under p_{obs} we simply conditioned by y which can be any of these values.

This is equivalent to

$$B_{k+1}^{n,y}(x_{k+1}) = \mathbb{E}[X_{k+1} + F_k^{n,Y}(X_k) - F_k^{n,Y}(X_{k+1})|X_{k+1} = x_{k+1}, Y = y],$$

Hence, we get that

$$B_{k+1}^n = \arg\min_{\mathrm{B}\in \mathrm{L}^2(\mathbb{R}^d\times\mathcal{Y},\mathbb{R}^d)} \mathbb{E}_{\bar{p}^n}[\|\mathrm{B}(X_{k+1}, Y) - (X_{k+1} + F_k^{n,Y}(X_k) - F_k^{n,Y}(X_{k+1}))\|^2],$$

which concludes the proof. □

E CONTINUOUS-TIME VERSIONS OF CSGM AND CDSB

In the following section, we consider the continuous-time version of CSGM and CDSB. The continuous-time dynamics we recover can be seen as the extensions of the continuous-time dynamics obtained in the unconditional setting, see Song et al. [2021], De Bortoli et al. [2021].

E.1 NOTATION

We start by introducing a few notations. The space of continuous functions from $[0, T]$ to $\mathbb{R}^d \times \mathcal{Y}$ is denoted $\mathcal{C} = \mathrm{C}([0, T], \mathbb{R}^d \times \mathcal{Y})$ and we denote $\mathscr{P}(\mathcal{C})$ the set of probability measures defined on $\mathcal{C}$. A probability measure $\mathbb{P} \in \mathscr{P}(\mathcal{C})$ is *associated with a diffusion* if it is a solution to a martingale problem, i.e. $\mathbb{P} \in \mathscr{P}(\mathcal{C})$ is associated with $\mathrm{d}\mathbf{X}_t = b(t, \mathbf{X}_t)\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ if for any $\varphi \in \mathrm{C}_c^2(\mathbb{R}^d, \mathbb{R})$, $(\mathbf{Z}_t^\varphi)_{t\in[0,T]}$ is a $\mathbb{P}$-local martingale, where for any $t \in [0, T]$

$$\mathbf{Z}_t^\varphi = \varphi(\mathbf{X}_t) - \textstyle\int_0^t \mathcal{A}_s(\varphi)(\mathbf{X}_s)\mathrm{d}s, \qquad \mathcal{A}_t(\varphi)(x) = \langle b(t, x), \nabla\varphi(x)\rangle + \Delta\varphi(x).$$

Here $\mathrm{C}_c^2(\mathbb{R}^d, \mathbb{R})$ denotes the space of twice differentiable functions from $\mathbb{R}^d$ to $\mathbb{R}$ with compact support. Doing so, $\mathbb{P}$ is uniquely defined up to the initial distribution $\mathbb{P}_0$. Finally, for any $\mathbb{P} \in \mathscr{P}(\mathcal{C})$, we introduce $\mathbb{P}^R$ the time reversal of $\mathbb{P}$, *i.e.* for any $\mathsf{A} \in \mathcal{B}(\mathcal{C})$ we have $\mathbb{P}^R(\mathsf{A}) = \mathbb{P}(\mathsf{A}^R)$ where $\mathsf{A}^R = \{t \mapsto \omega(T - t) \,:\, \omega \in \mathsf{A}\}$.

E.2 CONTINUOUS-TIME CSGM

Recall that in the unconditional setting, we consider a forward noising dynamics $(\mathbf{X}_t)_{t\in[0,T]}$ initialized with $\mathbf{X}_0 \sim p_{\text{data}}$ and satisfying the following Stochastic Differential Equation (SDE) $\mathrm{d}\mathbf{X}_t = -\mathbf{X}_t\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$, i.e. an Ornstein–Uhlenbeck process. In this case, under entropy condition on $(\mathbf{X}_t)_{t\in[0,T]}$ (see Cattiaux et al. [2021] for instance) we have that the time-reversal process $(\tilde{\mathbf{X}}_t)_{t\in[0,T]} = (\mathbf{X}_{T-t})_{t\in[0,T]}$ also satisfy an SDE given by $\mathrm{d}\tilde{\mathbf{X}}_t = \{\tilde{\mathbf{X}}_t + 2\nabla \log p_{T-t}(\tilde{\mathbf{X}}_t)\}\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$, where p_t is the density of $\mathbf{X}_t$ w.r.t. the Lebesgue measure, and $(\tilde{\mathbf{X}}_t)_{t\in[0,T]}$ is initialized with $\tilde{\mathbf{X}}_0 \sim \mathcal{L}(\mathbf{X}_T)$, the law of $\mathbf{X}_T$ of density q_T. Using the geometric ergodicity of the Ornstein–Uhlenbeck process, $\mathcal{L}(\mathbf{X}_T)$ is close (w.r.t. to the Kullback–Leibler divergence for instance) to $p_{\text{ref}} = \mathcal{N}(0, \mathrm{Id})$. Hence, we obtain that considering $(\mathbf{Z}_t)_{t\in[0,T]}$ such that $\mathbf{Z}_0 \sim \mathcal{N}(0, \mathrm{Id})$ and $\mathrm{d}\mathbf{Z}_t = \{\mathbf{Z}_t + 2\nabla \log p_{T-t}(\mathbf{Z}_t)\}\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$, $\mathbf{Z}_T$ is approximately distributed according to p_{data}. The Euler–Maruyama discretization of $(\mathbf{Z}_t)_{t\in[0,T]}$ is the SGM used in existing work.

In the conditional setting, we consider the following dynamics $\mathrm{d}\mathbf{X}_t = -\mathbf{X}_t\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ and $\mathrm{d}\mathbf{Y}_t = 0$, where $(\mathbf{X}_0, \mathbf{Y}_0) \sim p_{\text{join}}$. Note that we have $\mathbf{Y}_t = \mathbf{Y}_0$ for all $t \in [0, T]$. Using the ergodicity of the Ornstein–Uhlenbeck process, we get that $\mathcal{L}(\mathbf{X}_T, \mathbf{Y}_t)$ is close (w.r.t. to the Kullback–Leibler divergence for instance) to p_{jref}. Let $(\tilde{\mathbf{X}}_t, \tilde{\mathbf{Y}}_t)_{t\in[0,T]} = (\mathbf{X}_{T-t}, \mathbf{Y}_{T-t})_{t\in[0,T]}$. We have that $\mathrm{d}\tilde{\mathbf{X}}_t = \{\tilde{\mathbf{X}}_t + 2\nabla \log p_{T-t}(\tilde{\mathbf{X}}_t|\tilde{\mathbf{Y}}_t)\}\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ and $\mathrm{d}\tilde{\mathbf{Y}}_t = 0$ with $\tilde{\mathbf{X}}_0, \tilde{\mathbf{Y}}_0 \sim \mathcal{L}(\mathbf{X}_T, \mathbf{Y}_T)$. Hence, we obtain that considering $(\mathbf{Z}_t)_{t\in[0,T]}$ such that $(\mathbf{Z}_0, \mathbf{Y}_0) \sim p_{\text{jref}}$ and $\mathrm{d}\mathbf{Z}_t = \{\mathbf{Z}_t + 2\nabla \log p_{T-t}(\mathbf{Z}_t|\mathbf{Y}_0)\}\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$, $\mathbf{Z}_T$ is approximately distributed according to p_{data}. The Euler–Maruyama discretization of $(\mathbf{Z}_t, \mathbf{Y}_t)_{t\in[0,T]}$ is the conditional SGM.

E.3 CONNECTION WITH NORMALIZING FLOWS AND ESTIMATION OF THE EVIDENCE

It has been shown that SGMs can be used for log-likelihood computation. Here, we further show that they can be used to estimate the evidence $\log p(y^{\text{obs}})$ when $g(y^{\text{obs}}|x)$ can be computed pointwise. This is the case for many models considered in the diffusion literature, see for instance Kadkhodaie and Simoncelli [2021], Kawar et al. [2021, 2022]. Indeed, we have

that for any $x \in \mathbb{R}^d$, $\log p(y^{\text{obs}}) = \log g(y^{\text{obs}}|x) + \log p(x) - \log p(x|y^{\text{obs}})$. The term $\log p(x)$ can be estimated using an unconditional SGM whereas the term $\log p(x|y^{\text{obs}})$ can be estimated using a CSGM. Note that both conditional and unconditional SGM can be trained simultaneously adding a "sink" state to $\mathcal{Y}$, i.e. considering $\mathcal{Y} \cup \{\emptyset\}$, see Ho and Salimans [2021] for instance.

We briefly explain how one can compute $\log p(x|y^{\text{obs}})$ and refer to Song et al. [2021] for a similar discussion in the unconditional setting. Recall that the forward noising process is given by $\mathrm{d}\mathbf{X}_t = -\mathbf{X}_t \mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ and $\mathrm{d}\mathbf{Y}_t = 0$, where $(\mathbf{X}_0, \mathbf{Y}_0) \sim p_{\text{join}}$. We introduce another process $(\hat{\mathbf{X}}_t, \hat{\mathbf{Y}}_t)_{t\in[0,T]}$ with deterministic dynamics which has the same marginal distributions, i.e. $\mathcal{L}(\mathbf{X}_T, \mathbf{Y}_T) = \mathcal{L}(\hat{\mathbf{X}}_T, \hat{\mathbf{Y}}_T)$. This process is defined by $\mathrm{d}\hat{\mathbf{X}}_t = \{-\hat{\mathbf{X}}_t - \nabla \log p_t(\hat{\mathbf{X}}_t|\hat{\mathbf{Y}}_t)\}\mathrm{d}t$ and $\mathrm{d}\hat{\mathbf{Y}}_t = 0$ with $(\hat{\mathbf{X}}_0, \hat{\mathbf{Y}}_0) \sim p_{\text{join}}$. As one has $\mathrm{d}\log p_t(\hat{\mathbf{X}}_t|\hat{\mathbf{Y}}_t) = \mathrm{div}(-\hat{\mathbf{X}}_t - \nabla \log p_t(\hat{\mathbf{X}}_t|\hat{\mathbf{Y}}_t))\mathrm{d}t$, we can approximately compute $\log p(\hat{\mathbf{X}}_0|\hat{\mathbf{Y}}_0)$ by integrating numerically this Ordinary Differential Equation (ODE). There are practically three sources of errors, one is the score approximation, one is the numerical integration error and the last one one is due to the fact that $\mathcal{L}(\hat{\mathbf{X}}_T)$ is unknown so we use the approximation $\mathcal{L}(\hat{\mathbf{X}}_T) \approx p_{\text{ref}}$.

E.4 CONTINUOUS-TIME CDSB

In this section, we introduce an IPF algorithm for solving CSB problems in continuous-time. The following results are a generalization to the conditional framework of the continuous-time results of De Bortoli et al. [2021]. The CDSB algorithm described in Algorithm 1 can be seen as a Euler–Maruyama discretization of this IPF scheme combined to neural network approximations of the drifts. Let $\mathbb{P} \in \mathscr{P}(\mathcal{C})$ be a given reference measure (thought as the continuous time analog of $\bar{p}$). The dynamical continuous formulation of the SB problem can be written as follows

$$\Pi^\star = \arg\min \left\{ \mathrm{KL}(\Pi|\mathbb{P}) \,:\, \Pi \in \mathscr{P}(\mathcal{C}),\, \Pi_0 = p_{\text{join}},\, \Pi_T = p_{\text{jref}} \right\}.$$

We define the IPF $(\Pi^n)_{n\in\mathbb{N}}$ such that $\Pi^0 = \mathbb{P}$ and associated with $\mathrm{d}\mathbf{X}_t = -\mathbf{X}_t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ and $\mathrm{d}\mathbf{Y}_t = 0$, with $(\mathbf{X}_0, \mathbf{Y}_0) \sim p_{\text{join}}$. Next for any $n \in \mathbb{N}$ we define

$$\Pi^{2n+1} = \arg\min \left\{ \mathrm{KL}(\Pi|\Pi^{2n}) \,:\, \Pi \in \mathscr{P}(\mathcal{C}),\, \Pi_T = p_{\text{jref}} \right\},$$
$$\Pi^{2n+2} = \arg\min \left\{ \mathrm{KL}(\Pi|\Pi^{2n+1}) \,:\, \Pi \in \mathscr{P}(\mathcal{C}),\, \Pi_0 = p_{\text{join}} \right\}.$$

The following result is the continuous counterpart of Proposition 2.

Proposition 5. *Assume that $p_N, p_{\text{ref}} > 0$, $\mathrm{H}(p_{\text{ref}}) < +\infty$ and $\int_{\mathbb{R}^d} |\log p_{N|0}(x_N|x_0)| p_{\text{data}}(x_0) p_{\text{ref}}(x_N) < +\infty$. In addition, assume that there exist $\mathbb{M} \in \mathscr{P}(\mathcal{C})$, $U \in \mathrm{C}^1(\mathbb{R}^d, \mathbb{R})$, $C \geq 0$ such that for any $n \in \mathbb{N}$, $x \in \mathbb{R}^d$, $\mathrm{KL}(\Pi^n|\mathbb{M}) < +\infty$, $\langle x, \nabla U(x)\rangle \geq -C(1 + \|x\|^2)$ and $\mathbb{M}$ is associated with $(\mathbf{X}_t, \mathbf{Y}_t)_{t\in[0,T]}$ such that*

$$\mathrm{d}\mathbf{X}_t = -\nabla U(\mathbf{X}_t)\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t, \qquad \mathrm{d}\mathbf{Y}_t = 0 \tag{17}$$

with $\mathbf{X}_0$ distributed according to the invariant distribution of (17)*. Then, for any $n \in \mathbb{N}$ we have:*

(a) $(\Pi^{2n+1})^R$ is associated with $(\mathbf{X}_t^{2n+1}, \mathbf{Y}_t^{2n+1})_{t\in[0,T]}$ such that $\mathrm{d}\mathbf{X}_t^{2n+1} = b_{T-t}^n(\mathbf{X}_t^{2n+1}, \mathbf{Y}_t^{2n+1})\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ and $\mathrm{d}\mathbf{Y}_t^{2n+1} = 0$ with $(\mathbf{X}_0^{2n+1}, \mathbf{Y}_0^{2n+1}) \sim p_{\text{jref}}$;

(b) Π^{2n+2} is associated with $\mathrm{d}\mathbf{X}_t^{2n+2} = f_t^{n+1}(\mathbf{X}_t^{2n+2}, \mathbf{Y}_t^{2n+2})\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{B}_t$ with $(\mathbf{X}_0^{2n+2}, \mathbf{Y}_0^{2n+2}) \sim p_{\text{join}}$;

where for any $n \in \mathbb{N}$, $t \in [0,T]$, $x \in \mathbb{R}^d$ and $y \in \mathcal{Y}$, $b_t^n(x,y) = -f_t^n(x,y) + 2\nabla \log p_t^n(x|y)$, $f_t^{n+1}(x,y) = -b_t^n(x,y) + 2\nabla \log q_t^n(x|y)$, with $f_t^0(x) = -x$, and $p_t^n(\cdot|y)$, $q_t^n(\cdot|y)$ the densities of $\Pi_{t|y}^{2n}$ and $\Pi_{t|y}^{2n+1}$.

Proof. The proof of this proposition is a straightforward extension of [De Bortoli et al., 2021, Proposition 6]. □

We have seen in Appendix E.3 that it is possible to use CSGM to evaluate numerically the evidence when $g(y^{\text{obs}}|x)$ can be computed pointwise. The same strategy can be applied to both DSB and CDSB; see [De Bortoli et al., 2021, Section H.3] for details for DSB. In both cases, there exists an ordinary differential equation admitting the same marginals as the diffusion solving the SB, resp. the CSB, problem. By integrating these ODEs, we can obtain $\log p(x)$ and $\log p(x|y^{\text{obs}})$ for any x and thus can compute the evidence. Contrary to SGM and CSGM, the terminal state of the diffusion is exactly equal to the reference measure by design. So practically, we only have two instead of three sources of errors for SGM/CSGM: one is the drift approximation, one is the numerical integration error.

F FORWARD-BACKWARD SAMPLING

We detail in this section the forward-backward sampling approach and its connection with Spantini et al. [2022] when using an unconditional p_{ref}. In Spantini et al. [2022], it is proposed to first learn a deterministic transport map $\mathcal{U}(x, y) : \mathcal{X}\times\mathcal{Y} \to \mathcal{X}\times\mathcal{Y}$ from $(X, Y) \sim p_{\text{join}}$ to p_{jref}, then transport back the X-component through $\mathcal{S}(\cdot, y^{\text{obs}})^{-1}$ where $\mathcal{S} : \mathcal{X} \times \mathcal{Y} \to \mathcal{X}$ is the X-component of $\mathcal{U}$. In other words, this is to say sampling $\hat{X}^{\text{pos}} \sim p(x|y^{\text{obs}})$ corresponds to the two-step transformation

$$\hat{X}^{\text{ref}}, \hat{Y}^{\text{ref}} = \mathcal{U}(X, Y), \quad \hat{X}^{\text{pos}} = \mathcal{S}(\cdot, y^{\text{obs}})^{-1}(\hat{X}^{\text{ref}}). \tag{18}$$

The proposed CSB (5) can be thought of as the SB version of this idea. We learn a stochastic transport map from $p_{\text{join}}(x, y)$ to $p_{\text{ref}}(x, y)$. The CSB $\pi^\star$ defines, when conditioned on x_0 and y^{obs}, a (stochastic) transport map $\pi^{c,\star}_{y^{\text{obs}}}(x_N|x_0)$ from $p(x_0|y^{\text{obs}})$ to $p_{\text{ref}}(x_N)$; and, when conditioned on x_N and y^{obs}, a (stochastic) transport map $\pi^{c,\star}_{y^{\text{obs}}}(x_0|x_N)$ from $p_{\text{ref}}(x_N)$ to $p(x_0|y^{\text{obs}})$. In practice, we learn using CDSB separate half-bridges $\bar{p}^L(x_{1:N}|x_0, y^{\text{obs}})$ and $\bar{q}^L(x_{0:N-1}|x_N, y^{\text{obs}})$.

Spantini et al. [2022] remarked that, since the estimator $\mathcal{S}$ may be imperfect, $\hat{X}^{\text{ref}}$ may not have distribution p_{ref} exactly. In this case, (18) allows for the cancellation of errors between $\mathcal{S}$ and $\mathcal{S}(\cdot, y^{\text{obs}})^{-1}$.

We can exploit a similar idea in the CSB framework by defining an analogous forward-backward sampling procedure

$$\hat{X}_N \sim \bar{p}^L_{N|0}(x_N|X, Y), \quad \hat{X}_0 \sim \bar{q}^L_{0|N}(x_0|\hat{X}_N, y^{\text{obs}}). \tag{19}$$

As $\bar{q}^L$ is the approximate time reversal of $\bar{p}^L$, (19) exhibits similar advantages as (18) when the half-bridge $\bar{p}^L(x_{0:N}|y^{\text{obs}})$ is only an approximation to the CSB solution. While the forward and backward processes are stochastic and are not exact inverses of each other, using this forward-backward sampling may inevitably lead to increased variance. However, we found in practice that this forward-backward sampling procedure can still improve sampling quality (see *e.g.* Figures 2, G.4).

G EXPERIMENTAL DETAILS

G.1 EXPERIMENTAL SETUP

Network parameterization. Two parameterizations are possible for learning $\mathbf{F}$ and $\mathbf{B}$. In the main text, we described one parameterization in which we parameterize $\mathbf{F}, \mathbf{B}$ directly as $\mathbf{F}^y_\phi(k, x), \mathbf{B}^y_\theta(k, x)$ and learn the network parameters ϕ, θ. Alternatively, we can parameterize $\mathbf{F}^y(k, x) = x + \gamma_{k+1}\mathbf{f}^y_\phi(k, x), \mathbf{B}^y(k + 1, x) = x + \gamma_{k+1}\mathbf{b}^y_\theta(k + 1, x)$ and learn the network parameters ϕ, θ for $\mathbf{f}^y_\phi, \mathbf{b}^y_\theta$ instead. For the 2D and BOD examples, we use a fully connected network with positional encodings as in De Bortoli et al. [2021] to learn $\mathbf{f}^y_\phi, \mathbf{b}^y_\theta$, with y as an additional input by concatenation with x. For the MNIST and CelebA examples, we follow earlier work and utilize the conditional U-Net architecture in Dhariwal and Nichol [2021]. Since residual connections are already present in the U-Net architecture, we can adopt the $\mathbf{F}^y_\phi, \mathbf{B}^y_\theta$ parameterization. In our experiments, we experiment with both parameterizations and find that the $\mathbf{f}^y_\phi, \mathbf{b}^y_\theta$ parameterization is more suitable for neural network architectures without residual connections. On the other hand, both parameterizations obtained good results when using the U-Net architecture. For consistency, all reported image experiment results use the $\mathbf{F}^y_\phi, \mathbf{B}^y_\theta$ parameterization, and we leave the choice of optimal parameterization as future research.

Network warm-starting. As observed by De Bortoli et al. [2021], since the networks at IPF iteration n are close to the networks at iteration $n - 1$, it is possible to warm-start ϕ^n, θ^n at ϕ^{n-1}, θ^{n-1} respectively. Empirically, we observe that this approach can significantly reduce training time at each CDSB iteration. Compared to CSGM, we usually observe immediate improvement in $\mathbf{B}_{\theta^2}$ during CDSB iteration 2 when the network is warm-started at θ^1 after CDSB iteration 1 (see *e.g.* Figure G.1). As CSGM corresponds to the training objective of θ^1 at CDSB iteration 1, this shows that the CDSB framework is a generalization of CSGM with observable benefits starting CDSB iteration 2.

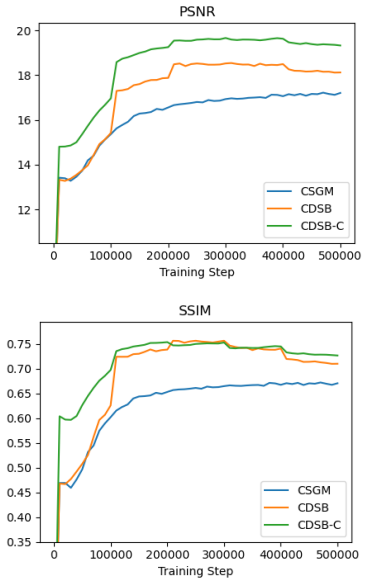


Figure G.1: Test set PSNR and SSIM against the number of training steps for MNIST 4x SR.

		MCMC	CDSB	CDSB-FB	CDSB-C	MGAN	IT
Mean	x_1	.075	.066±.010	.068±.010	**.072±.007**	.048	.034
	x_2	.875	.897±.019	.897±.017	**.891±.013**	.918	.902
Var	x_1	.190	.184±.007	**.190±.007**	.188±.005	.177	.206
	x_2	.397	.387±.006	.391±.006	**.393±.005**	.419	.457
Skew	x_1	1.94	**1.90±.038**	2.01±.041	**1.90±.028**	1.83	1.63
	x_2	.681	.591±.018	.628±.018	.596±.014	**.630**	.872
Kurt	x_1	8.54	7.85±.210	**8.54±.239**	8.00±.147	7.64	7.57
	x_2	3.44	3.33±.035	**3.51±.041**	3.27±.035	3.19	3.88

Table G.1: Estimated posterior moments and their standard deviations for the BOD example. The closest estimates to MCMC are highlighted in bold.

Conditional initialization. In the main text, we considered joint reference measures of the form $p_{\text{jref}}(x, y) = p_{\text{ref}}(x|y)p_{\text{obs}}(y)$ and simple choices for $p_{\text{ref}}(x|y)$ such as $\mathcal{N}(x; y, \sigma^2_{\text{ref}}\,\text{Id})$ for image super-resolution. We also explore two more choices for $p_{\text{ref}}(x|y)$ in our experiments. The first choice simply replaces the initialization mean from y to a neural network function $\mu_{\text{ref}}(y)$. This neural network can be pre-trained directly to estimate the conditional mean of $p(x|y)$ using standard regression with MSE loss. In the case of multi-modal $p(x|y)$ such as in the case of image inpainting, we can also train $\mu_{\text{ref}}(y)$ to estimate the conditional mean of $p_N(x_N|y)$, where x_N follows a standard diffusion process. In essence, we can train $\mu_{\text{ref}}(y)$ to facilitate $p_N(x_N|y) \approx p_{\text{ref}}(x_N|y)$ and shorten the noising process. Note that the CDSB framework is still useful in this context since $p_N(x_N|y)$ may not be well-approximated by a Gaussian distribution, which is precisely the issue CDSB is designed to tackle. Another class of conditional initialization we consider is the Ensemble Kalman Filter (EnKF), which is an ensemble-based method approximating linear Gaussian posterior updates. In this case, $p_{\text{ref}}(x|y)$ is taken to be $\mathcal{N}(x; \mu_{\text{ref}}(y), \text{diag}(\sigma^2_{\text{ref}}(y))$ where $\mu_{\text{ref}}(y), \sigma^2_{\text{ref}}(y)$ are the sample mean and variance of the EnKF posterior ensemble. Intuitively, $p_{\text{ref}}(x|y)$ is now an approximation of the true posterior $p(x|y)$ using linear prior-to-posterior mappings, which is further corrected for non-linearity and non-Gaussianity by the CDSB.

Time step schedule. For the selection of the time step sequence $\{\gamma_k\}_{k=1}^N$, we follow Ho et al. [2020], Dhariwal and Nichol [2021] and consider a linear schedule where $\gamma_1 = \gamma_{\text{min}}$, $\gamma_N = \gamma_{\text{max}}$, and $\gamma_k = \gamma_{\text{min}} + \frac{k-1}{N-1}(\gamma_{\text{max}} - \gamma_{\text{min}})$. In this way, the diffusion step size gets finer as the reverse process approaches $\pi_0 = p_{\text{data}}$, so as to increase the accuracy of the generated samples.

G.2 2D SYNTHETIC EXAMPLES

For the 2D examples, we use $N = 50$ diffusion steps and choose the time step schedule such that $\gamma_{\text{min}} = 10^{-4}, \gamma_{\text{max}} = 0.005$. At each IPF iteration, we train the network for 30,000 iterations using the Adam optimizer with learning rate 10^{-4} and a batch size of 100.

G.3 BIOCHEMICAL OXYGEN DEMAND MODEL

For the BOD example, we again use $N = 50$ diffusion steps with time schedule $\gamma_{\text{min}} = \gamma_{\text{max}} = 0.01$. For CDSB-C, we use the shortened time schedule $\gamma_{\text{min}} = \gamma_{\text{max}} = 0.005$ and a neural network regressor of the same architecture (with x and k components removed) as the conditional initialization. The batch size and optimizer settings are the same as above.

We report the estimated posterior moments as well as their standard deviation in Table G.1. We further plot the convergence of RMSE for each of the statistics in Figure G.2. As can be observed, IPF converges after about 20 iterations, and errors for all statistics are improved compared with CSGM (corresponding to IPF iteration 1). Using conditional initialization also helps with localizing the problem and reduces estimation errors especially in early iterations.

G.4 IMAGE EXPERIMENTS

For all image experiments, we use the Adam optimizer with learning rate 10^{-4} and train for 500k iterations in total. Since both $\mathbf{F}$ and $\mathbf{B}$ needs to be trained, the training time is approximately doubled for CDSB. Following Song and Ermon [2020], we make use of the exponential moving average (EMA) of the network parameters with EMA rate 0.999 at test time. We use $\gamma_{\text{min}} = 5 \times 10^{-5}$ for all experiments unless indicated otherwise and perform a parameter sweep for γ_{max} in

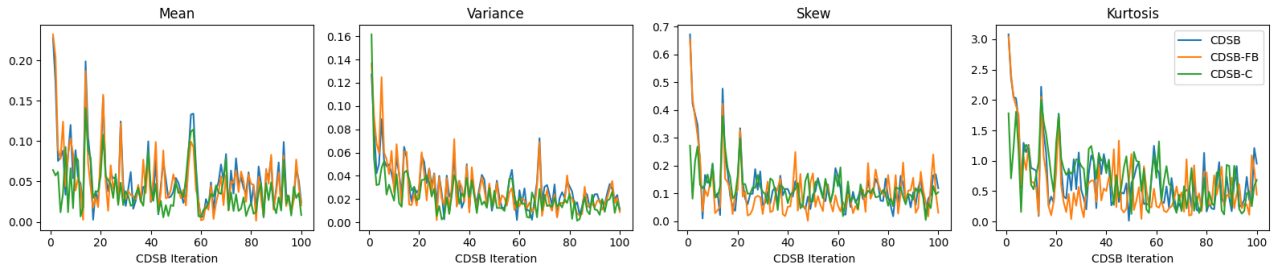


Figure G.2: Convergence of estimated posterior moments with increasing number of CDSB iterations.

$\{0.005, 0.01, 0.05, 0.1\}$. The optimal γ_{max} depends on the number of timesteps N and the discrepancy between $p(x|y)$ and p_{ref}. When using large N or conditional $p_{\text{ref}}(x|y)$, we find γ_{max} can be taken smaller.

G.4.1 MNIST

For the MNIST dataset, we use a U-Net architecture with 3 resolution levels each with 2 residual blocks. The numbers of filters at each resolution level are 64, 128, 128 respectively. The total number of parameters is 6.6m, and we use batch size 128 for training. Since we observe overfitting on the MNIST training set for all methods, we also apply dropout with $p = 0.1$ for the MNIST experiments. For each CDSB iteration, 100k or 250k training steps are used, corresponding to $L = 5$ or $L = 2$ CDSB iterations in total, which we find to be sufficient on this simpler dataset.

For $N = 10$, CDSB generates a minibatch of 100 images in approximately 0.8 seconds when run on a GTX 1080Ti. As a baseline comparison, we experimented with the methodology in Kadkhodaie and Simoncelli [2021] on the same MNIST test set and find that it gives PSNR/SSIM values of 15.78/0.72 and 12.49/0.47 for super-resolution and inpainting respectively (*c.f.* Table 2). Around 250 iterations are required for generating each image, or approximately 1 second generation time for 1 image on a GTX 1080Ti. In comparison, the CDSB methodology is much more efficient and achieves better image quality on both tasks.

G.4.2 CelebA 64x64

For the CelebA dataset, we use a U-Net architecture with 4 resolution levels each with 2 residual blocks and self-attention blocks at 16×16 and 8×8 resolutions. The numbers of filters at each resolution level are 128, 256, 256, 256 respectively. The total number of parameters is 39.6m, and we use batch size 128 for training. For each CDSB iteration, 10k or 25k training steps are used, corresponding to $L = 50$ or $L = 20$ CDSB iterations in total. For smaller γ_{max}, we find that higher number of CDSB iterations are beneficial.

For $N = 20, 50$, CDSB generates a minibatch of 100 images in approximately 12, 30 seconds when run on a Titan RTX. As a baseline comparison, we find that CDSB-C with $N = 20$ even outperforms a standard CSGM with $N = 200$, which achieves PSNR/SSIM values of approximately 20.98/0.62. To ensure that conditional initialization is not the sole contributor to the gain in sample quality, we further compare CDSB-C ($N = 50$) to a CSGM ($N = 50$) with conditional initialization. The forward noising process is also modified to the discretized Ornstein–Uhlenbeck process targeting $p_{\text{ref}}(x|y)$ as described in Section 5.2. This modification achieved PSNR/SSIM values of 20.84/0.59 (*c.f.* Table 2), which indicates that the CDSB framework presents larger benefits in addition to conditional initialization.

As another baseline comparison, the SNIPS algorithm [Kawar et al., 2021] reports PSNR of 21.90 for 8 CelebA test images and, when averaging across 8 predicted samples for each of the images, a PSNR of 24.31. The algorithm requires 2500 iterations for image generation, or approximately 2 minutes for producing 8 samples when run on an RTX 3080 as reported by Kawar et al. [2021]. On the same test benchmark, CDSB with $N = 50$ achieved PSNR values of 21.87 and 24.20 respectively in 3.1 seconds, thus achieving similar levels of sample quality using much less iterations. Furthermore, the SNIPS algorithm is applicable specifically for tractable linear Gaussian inverse problems, whereas CDSB is more general and does not rely on tractable likelihoods.

G.4.3 CelebA 160x160

We adopt the official implementation and pre-trained checkpoints of SRFlow[3] and make use of a higher resolution version of CelebA (160x160) following Lugmayr et al. [2020] in only Section 7.3.2. For CSGM and CDSB, we use a U-Net architecture with 4 resolution levels each with 2 residual blocks. The numbers of filters at each resolution level are 128, 256, 256, 512 respectively. The total number of parameters is 71.0m while SRFlow has total number of parameters 40.0m. We use a batch size of 32 for training the CSGM and CDSB models.

When $p_{\text{ref}}(x|y)$ is defined by SRFlow, it is infeasible to use a discretized Ornstein–Uhlenbeck process targeting $p_{\text{ref}}(x|y)$ as in Section 5.2. We instead use a discretized Brownian motion for $p_{k+1|k}$, or equivalently the Variance Exploding (VE) SDE [Song and Ermon, 2019, Song et al., 2021]. This has the interpretation as a entropy regularized Wasserstein-2 optimal transport problem as discussed in Section 3.2, i.e. CDSB-C seeks to minimize the total squared transport distance between SRFlow $p_{\text{ref}}(x|y)$ and the true posterior $p(x|y)$. We use the time schedule $\gamma_{\text{min}} = \gamma_{\text{max}} = 0.005$ with comparatively higher γ_{min} in order to accelerate convergence under $N = 10$ timesteps. We provide additional samples from SRFlow, CDSB-C as well as CSGM-C in Figures G.8, G.9, G.10.

G.5 OPTIMAL FILTERING IN STATE-SPACE MODELS

For the sake of completeness, we first give details of the Lorenz-63 model here. It is defined for $x \in \mathbb{R}^3$ under the following ODE system

$$\frac{\mathrm{d}x[1]}{\mathrm{d}\tau} = \sigma(x[2] - x[1]), \quad \frac{\mathrm{d}x[2]}{\mathrm{d}\tau} = x[1](\rho - x[3]) - x[2], \quad \frac{\mathrm{d}x[3]}{\mathrm{d}\tau} = x[1]x[2] - \theta x[3].$$

We consider the values $\sigma = 10$, $\rho = 28$ and $\theta = 8/3$, which results in chaotic dynamics famously known as the Lorenz attractor. We integrate this system using the 4th order Runge–Kutta method with step size 0.05. For the state-space model, we define $(X_t)_{t\geq 1}$ as the states $(x[1], x[2], x[3])$ of the system at regular intervals of $\delta\tau = 0.1$ with small Gaussian perturbations of mean 0 and variance 10^{-4}, and $(Y_t)_{t\geq 1}$ as noisy observations of $(X_t)_{t\geq 1}$ with Gaussian noise of mean 0 and variance 4. More explicitly, the transition density is thus defined for $x_t = (x_t[1], x_t[2], x_t[3]) \in \mathbb{R}^3$ as

$$f(x_t|x_{t-1}) = \mathcal{N}(x_t; \text{RK4}(x_{t-1}, 0.1), 10^{-4}\,\text{Id}), \quad g(y_t|x_t) = \mathcal{N}(y_t; x_t, 4\,\text{Id}),$$

where $\text{RK4}(x_t, 0.1)$ is the 4th order Runge–Kutta operator (with step size 0.05) for the Lorenz-63 dynamics with initial condition x_t and termination time 0.1.

We run the model for 4,000 time steps and perform Bayesian filtering for the last 2,000 time steps. To accelerate the sequential inference process, we use linear regression in this example to fit $\mathbf{F}, \mathbf{B}$ with nonlinear feature expansion using radial basis functions. Similar to Spantini et al. [2022], we experiment with the number of nonlinear features from 1 to 3 RBFs, in addition to the linear feature. We find that as the ensemble size M increases, increasing the number of features is helpful for lowering filtering errors, suggesting that bias-variance tradeoff is at play.

Since the system's dynamics are chaotic and can move far from the origin and display different scaling for each dimension, it is not suitable to choose $p_{\text{ref}}(x) = \mathcal{N}(x; 0, \text{Id})$. Therefore, for CSGM and CDSB, we let $p_{\text{ref}}(x) = \mathcal{N}(x; \mu_{\text{ref}}, \text{diag}(\sigma^2_{\text{ref}}))$ where $\mu_{\text{ref}}, \sigma^2_{\text{ref}}$ are the estimated mean and variance of the prior predictive distribution $p(x_t|y^{\text{obs}}_{1:t-1})$ at time t. For CSGM-C and CDSB-C, we let $p_{\text{ref}}(x|y) = \mathcal{N}(x; \mu_{\text{ref}}, \text{diag}(\sigma^2_{\text{ref}}))$ where the estimated posterior mean and variance are returned by EnKF. Furthermore, we scale the diffusion process's time step dimensionwise by the variance of the reference measure σ^2_{ref}. We consider a short diffusion process with $N = 20$, and a long diffusion process with $N = 100$. We let $\gamma_{\text{min}} = 0.0005 \cdot \sigma^2_{\text{ref}}$ and $\gamma_{\text{max}} = 0.05 \cdot \sigma^2_{\text{ref}}$ for the short diffusion process, and reduce γ_{max} by a half for the long diffusion process.

We report the RMSEs between each algorithm's filtering means and the ground truth filtering means in Table 4. We compute the ground truth filtering means using a particle filter with $M = 10^6$ particles. In addition, we report the RMSEs between each algorithm's filtering means and the true states $x_{1:T}$ in Table G.2a, and between each algorithm's filtering standard deviations and the ground truth standard deviations in Table G.2b. Similarly, we observe that CDSB and CDSB-C achieve lower errors than CSGM and EnKF. Interestingly, CSGM-C performs similarly well as CDSB-C for state estimation when $N = 100$ steps, but performs worse for standard deviation estimation. In the case where the ensemble size $M = 200$, however, when using the long diffusion process we observe occasional large errors for CDSB and CDSB-C. We conjecture that since CDSB is an iterative algorithm, inevitably small errors in regression can be accumulated. For small ensemble size

[3] https://github.com/andreas128/SRFlow

M	200	500	1000	2000
EnKF	.476±.010	.474±.005	.475±.005	.475±.003
CSGM (short)	Diverges			
CDSB (short)	.464±.013	.391±.010	.369±.007	.352±.008
CSGM-C (short)	Diverges			
CDSB-C (short)	**.428±.016**	**.378±.012**	**.359±.015**	**.340±.007**
CSGM (long)	**.431±.010**	.376±.008	.360±.012	.343±.006
CDSB (long)	.582±.328	.370±.012	.348±.006	.333±.006
CSGM-C (long)	.434±.057	**.367±.011**	.346±.008	.336±.004
CDSB-C (long)	.660±.310	.368±.016	**.344±.010**	**.331±.006**

(a)

M	200	500	1000	2000
EnKF	.255±.003	.286±.002	.296±.001	.300±.003
CSGM (short)	Diverges			
CDSB (short)	.203±.005	.167±.003	.150±.002	.137±.002
CSGM-C (short)	Diverges			
CDSB-C (short)	**.148±.004**	**.124±.002**	**.108±.002**	**.099±.001**
CSGM (long)	.204±.005	.163±.008	.140±.002	.129±.001
CDSB (long)	**.140±.008**	.129±.003	.123±.003	.120±.002
CSGM-C (long)	.186±.005	.142±.003	.120±.001	.109±.002
CDSB-C (long)	.176±.006	**.120±.002**	**.110±.003**	**.106±.002**

(b)

Table G.2: RMSEs over 10 runs between (a) each algorithm's filtering means and the true states $x_{1:T}$ for $N = 20$ (short) and $N = 100$ (long); (b) each algorithm's filtering standard deviations and the ground truth filtering standard deviations. The lowest errors are highlighted in bold.

and large number of diffusion steps, the model may thus be more prone to overfitting. However, for larger ensemble size $M \geq 500$ we do not observe this issue.

References

Patrick Cattiaux, Giovanni Conforti, Ivan Gentil, and Christian Léonard. Time reversal of diffusion processes under a finite entropy condition. *arXiv preprint arXiv:2104.07708*, 2021.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion Schrödinger bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems*, 2021.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat GAN on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Zahra Kadkhodaie and Eero P Simoncelli. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. In *Advances in Neural Information Processing Systems*, 2021.

Bahjat Kawar, Gregory Vaksman, and Michael Elad. SNIPS: Solving noisy inverse problems stochastically. In *Advances in Neural Information Processing Systems*, 2021.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *arXiv preprint arXiv:2201.11793*, 2022.

Solomon Kullback. *Information Theory and Statistics*. Dover Publications, Inc., Mineola, NY, 1997. Reprint of the second (1968) edition.

Flavien Léger. A gradient descent perspective on Sinkhorn. *Applied Mathematics & Optimization*, 84(2):1843–1855, 2021.

Christian Léonard. Some properties of path measures. In *Séminaire de Probabilités XLVI*, pages 207–230. Springer, 2014.

Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In *ECCV*, 2020.

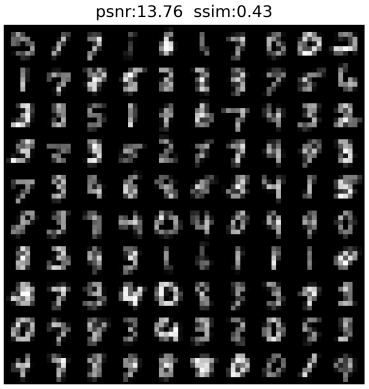
Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Foundations and Trends® in Machine Learning*, 11(5-6): 355–607, 2019.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. In *Advances in Neural Information Processing Systems*, 2020.

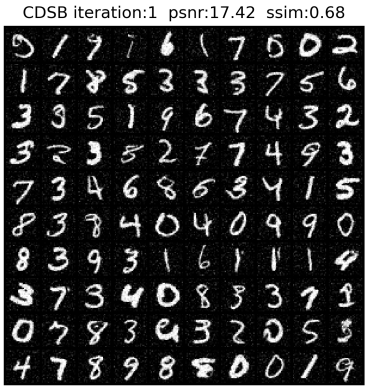
Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Alessio Spantini, Ricardo Baptista, and Youssef Marzouk. Coupling techniques for nonlinear ensemble filtering. *SIAM Review*, 2022. to appear.

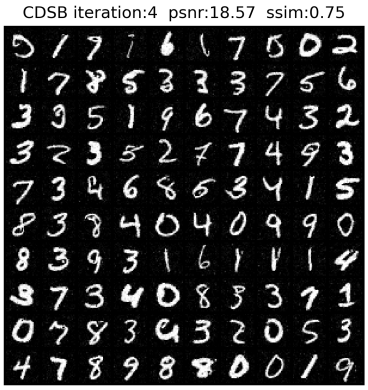


(a) y^{obs}

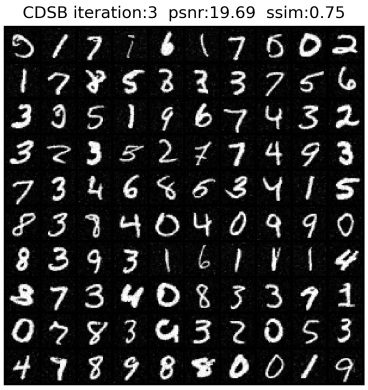
(b) Ground truth



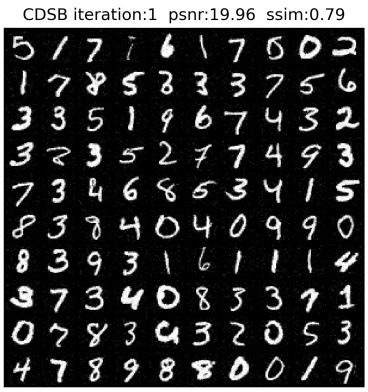
(c) CSGM $N = 5$



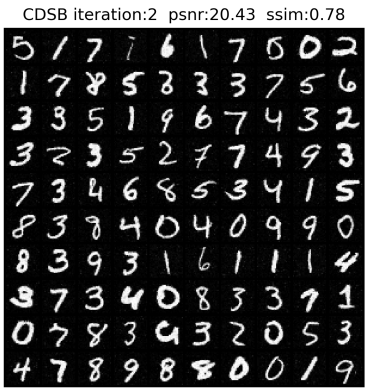
(d) CDSB $N = 5$



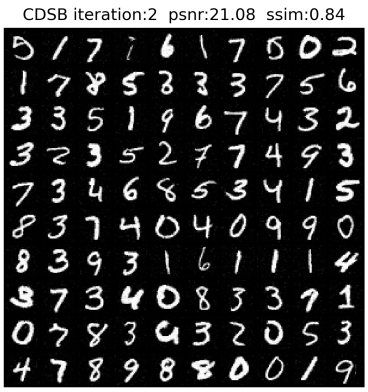
(e) CDSB-C $N = 5$



(f) CSGM $N = 10$

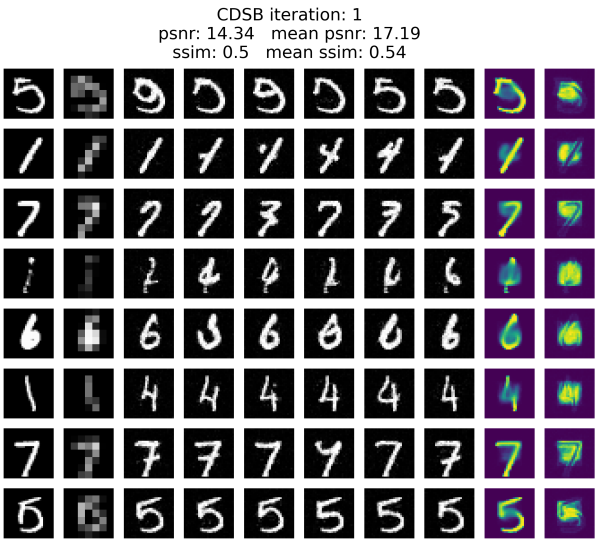


(g) CDSB $N = 10$

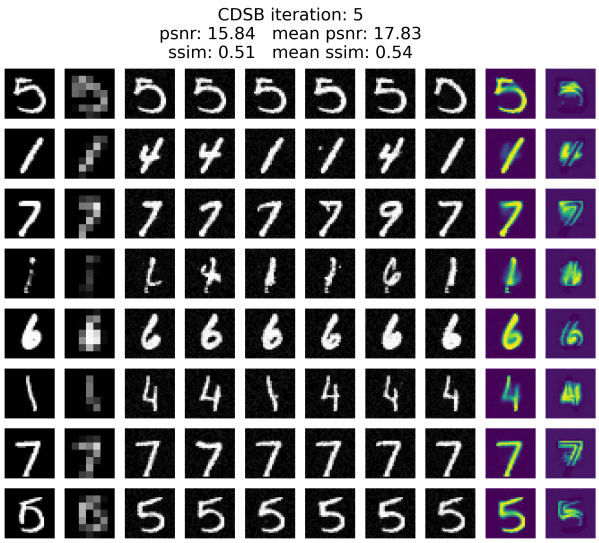


(h) CDSB-C $N = 10$

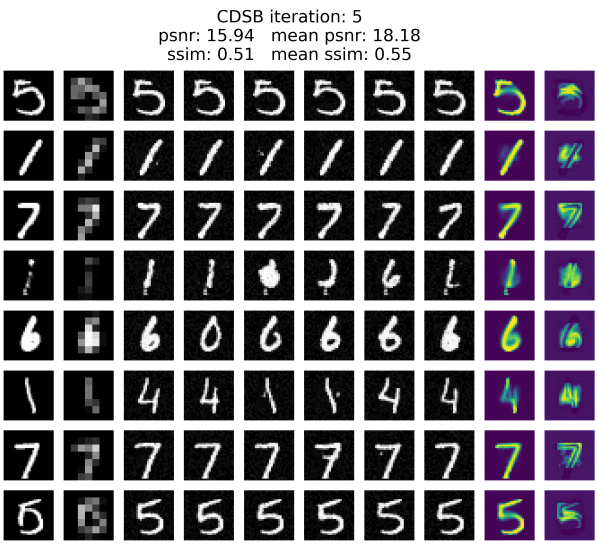
Figure G.3: Additional samples for the MNIST 4x SR task.



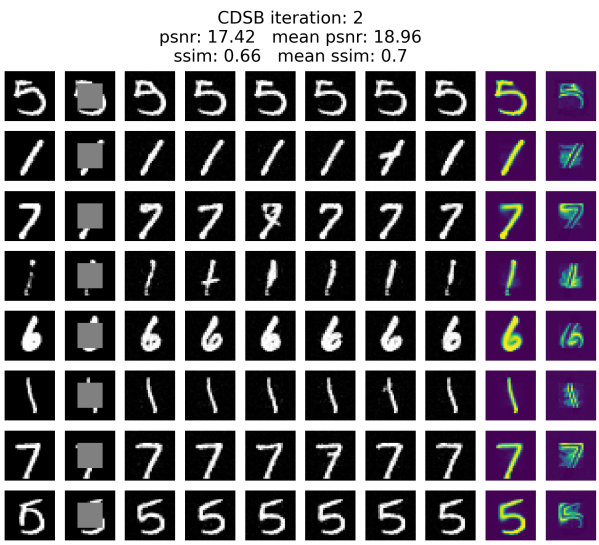
(a) CSGM $N = 10$



(b) CDSB $N = 10$



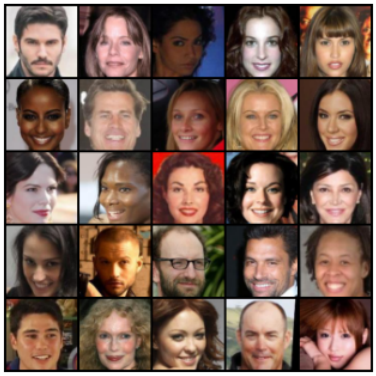
(c) CDSB-FB $N = 10$



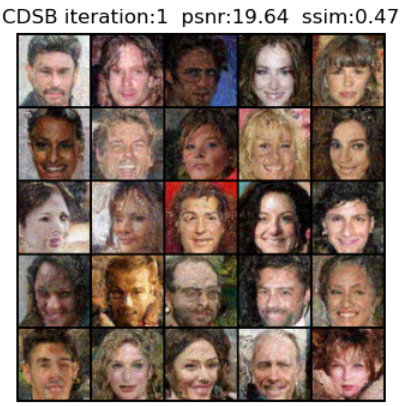
(d) CDSB-C $N = 10$

Figure G.4: Uncurated conditional samples for the MNIST 14x14 inpainting task. The first two columns correspond to ground truth, y^{obs}, and the last two columns correspond to the mean and standard deviation of 100 samples.

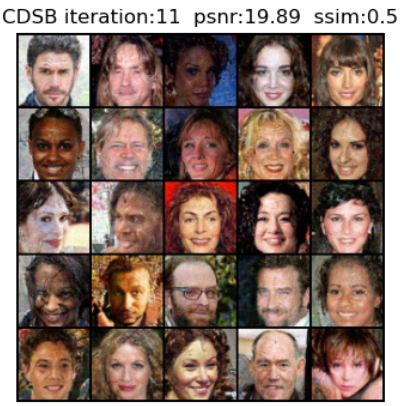
(a) y^{obs}



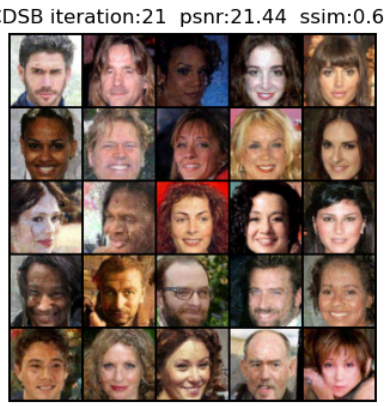
(b) Ground truth



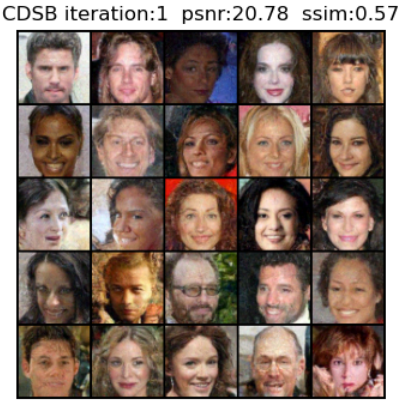
(c) CSGM $N = 20$



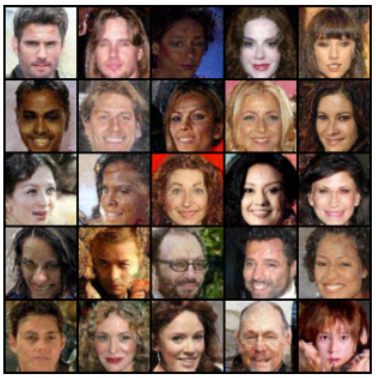
(d) CDSB $N = 20$



(e) CDSB-C $N = 20$



(f) CSGM $N = 50$

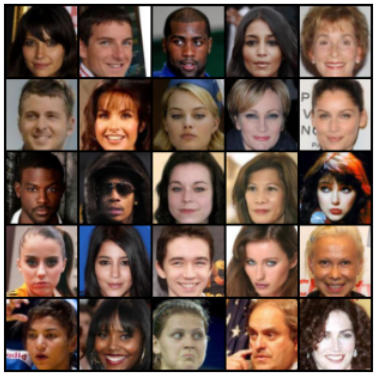


(g) CDSB $N = 50$

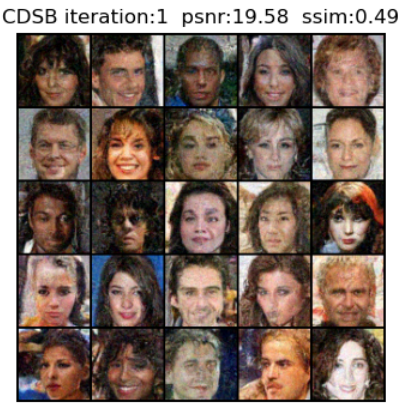
(h) CDSB-C $N = 50$

Figure G.5: Uncurated samples for the CelebA 4x SR with Gaussian noise task.

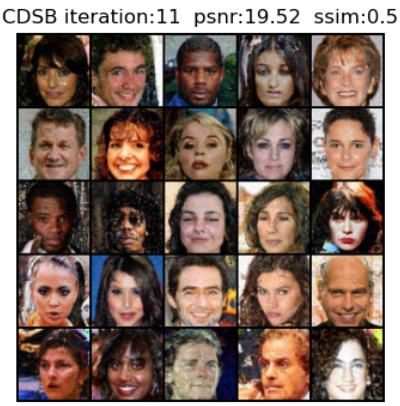
(a) y^{obs}



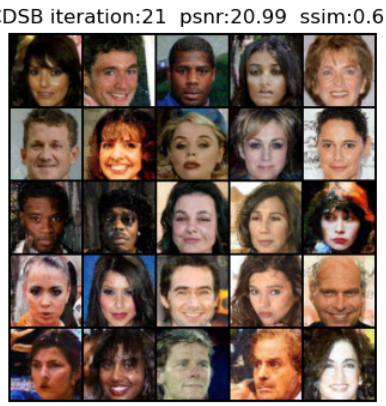
(b) Ground truth



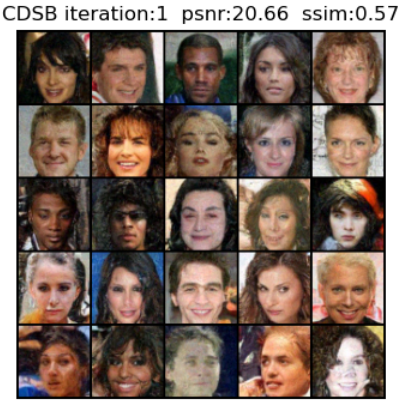
(c) CSGM $N = 20$



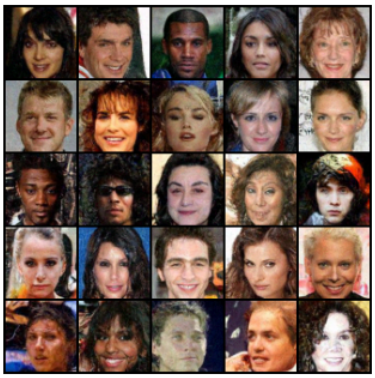
(d) CDSB $N = 20$



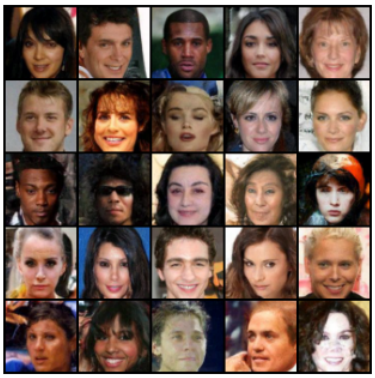
(e) CDSB-C $N = 20$



(f) CSGM $N = 50$



(g) CDSB $N = 50$



(h) CDSB-C $N = 50$

Figure G.6: Uncurated samples for the CelebA 4x SR with Gaussian noise task.

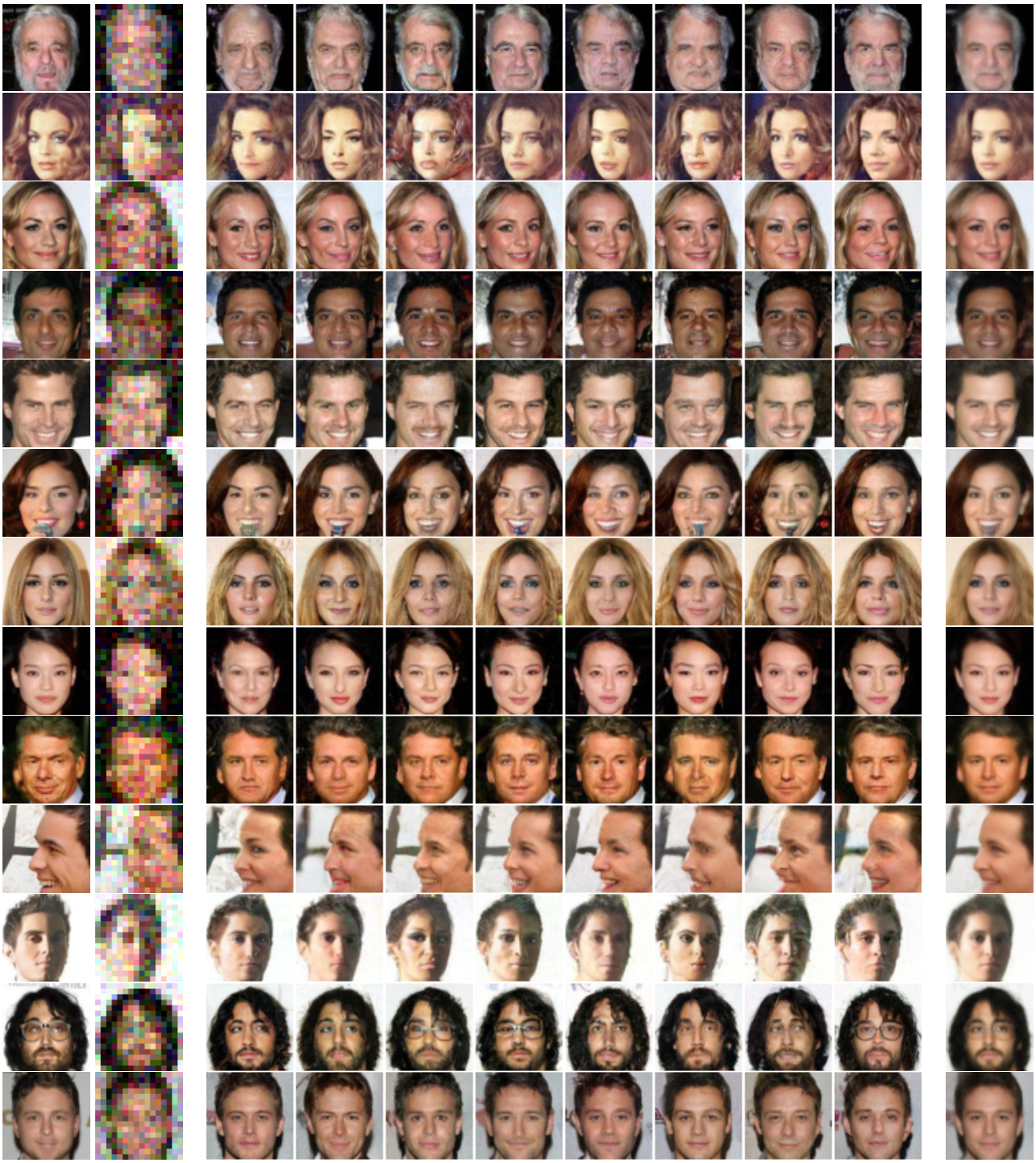
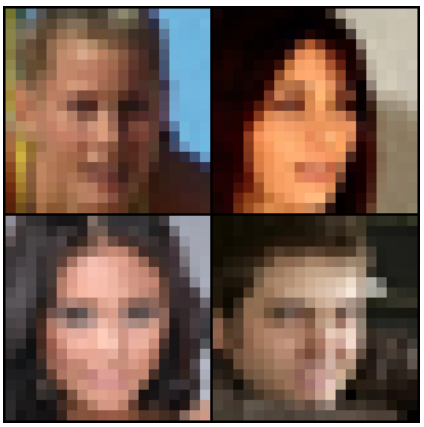


Figure G.7: Uncurated conditional samples using CDSB-C with $N = 50$ for the CelebA 4x SR with Gaussian noise task. The first two columns correspond to ground truth, y^{obs}, and the last column corresponds to the mean of the middle 8 samples.



(a) y^{obs}

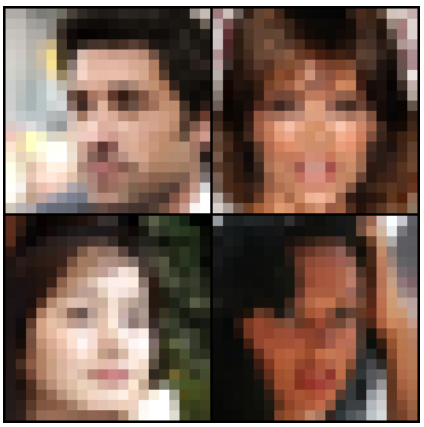
(b) Ground truth

(c) SRFlow

(d) CSGM-C $N = 10$

(e) CDSB-C $N = 10$

Figure G.8: Additional uncurated samples for the CelebA 8x SR task.



(a) y^{obs}

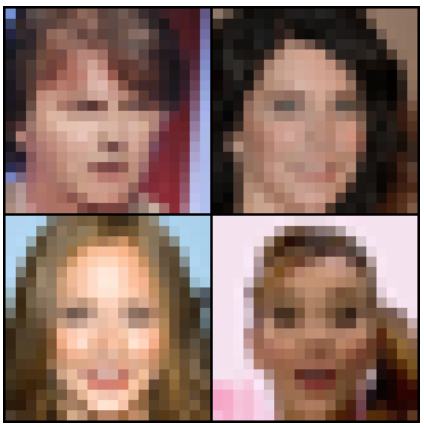
(b) Ground truth

(c) SRFlow

(d) CSGM-C $N = 10$

(e) CDSB-C $N = 10$

Figure G.9: Additional uncurated samples for the CelebA 8x SR task.



(a) y^{obs}

(b) Ground truth

(c) SRFlow

(d) CSGM-C $N = 10$

(e) CDSB-C $N = 10$

Figure G.10: Additional uncurated samples for the CelebA 8x SR task.