

## A APPENDIX

### A.1 PROOF OF THE INSTANTANEOUS CHANGE OF LOG-LIKELIHOOD ESTIMATE

**Proposition** (Instantaneous Change of Log-likelihood Estimate). *Let  $\mathbf{z}(t)$  be a finite continuous random variable at time  $t$  as the solution of a differential equation  $\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}(\mathbf{z}(t), t)$  with initial value  $\mathbf{z}(0) = \mathbf{x}$ . Assuming that  $\tilde{p}_0 = \mu$  at  $t = 0$  and  $\mathbf{f}$  is uniformly Lipschitz continuous in  $\mathbf{z}$  and  $t$ , then the change in estimated log-likelihood  $\log \tilde{p}_t(\mathbf{x})$  at  $t$  follows a differential equation:*

$$\frac{d \log \tilde{p}_t(\mathbf{x})}{dt} = \nabla \cdot \mathbf{f}(\mathbf{z}(t), t) + \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t).$$

*Proof.* To prove this theorem, we take the infinitesimal limit of finite changes of  $\log \tilde{p}_t(\mathbf{x})$  through time. As  $\mathbf{f}$  is assumed to be Lipschitz continuous in  $\mathbf{z}(t)$  and  $t$ ,  $\Phi_t(\mathbf{x})$  represents the unique solution of the ODE (eq. (2)) at time  $t$  on the initial value  $\mathbf{x}$ :

$$\Phi_t(\mathbf{x}) = \mathbf{x} + \int_0^t \mathbf{f}(\Phi_\tau(\mathbf{x}), \tau) d\tau,$$

We also denote the transformation on  $\mathbf{z}(t)$  over an  $\epsilon$  change in time as:

$$\mathbf{z}(t + \epsilon) = \Phi_\epsilon(\mathbf{z}(t)) = \Phi_{t+\epsilon}(\mathbf{x}).$$

Using the definition of estimated log density  $\tilde{p}_t(\mathbf{x})$  in eq. (4), the infinitesimal limit is:

$$\begin{aligned} \frac{d \log \tilde{p}_t(\mathbf{x})}{dt} &:= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} (\log |\det(\mathcal{J}_{\Phi_{t+\epsilon}}(\mathbf{x}))| - \log |\det(\mathcal{J}_{\Phi_t}(\mathbf{x}))| + \log \mu(\Phi_{t+\epsilon}(\mathbf{x})) - \log \mu(\Phi_t(\mathbf{x}))) \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} (\log |\det(\mathcal{J}_{\Phi_{t+\epsilon}}(\mathbf{x}))| - \log |\det(\mathcal{J}_{\Phi_t}(\mathbf{x}))|) + \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} (\log \mu(\Phi_{t+\epsilon}(\mathbf{x})) - \log \mu(\Phi_t(\mathbf{x}))). \end{aligned}$$

The derivation of first term is very similar to (Chen et al., 2018, theorem 1) except the sign of the function:

$$\begin{aligned} &\lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} (\log |\det(\mathcal{J}_{\Phi_{t+\epsilon}}(\mathbf{x}))| - \log |\det(\mathcal{J}_{\Phi_t}(\mathbf{x}))|) \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} (\log |\det(\mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t)))| + \log |\det(\mathcal{J}_{\Phi_t}(\mathbf{x}))| - \log |\det(\mathcal{J}_{\Phi_t}(\mathbf{x}))|) \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \log |\det(\mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t)))|, \end{aligned}$$

where we summarize the main steps:

$$\begin{aligned} \lim_{\epsilon \rightarrow 0^+} \frac{\log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|}{\epsilon} &= \lim_{\epsilon \rightarrow 0^+} \frac{\frac{\partial}{\partial \epsilon} \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|}{\frac{\partial \epsilon}{\partial \epsilon} \rightarrow 1} && \text{(L'Hopital's rule)} \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{\frac{\partial}{\partial \epsilon} |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|}{|\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))| \rightarrow 1} \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{\partial}{\partial \epsilon} |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))| \\ &= \lim_{\epsilon \rightarrow 0^+} \text{Tr} \left( \text{adj} \left( \frac{\partial}{\partial \mathbf{z}(t)} \Phi_\epsilon(\mathbf{z}(t)) \right) \frac{\partial}{\partial \epsilon} \frac{\partial}{\partial \mathbf{z}(t)} \Phi_\epsilon(\mathbf{z}(t)) \right) && \text{(Jacobi's formula)} \\ &= \text{Tr} \left( \lim_{\epsilon \rightarrow 0^+} \frac{\partial}{\partial \epsilon} \frac{\partial}{\partial \mathbf{z}(t)} \Phi_\epsilon(\mathbf{z}(t)) \right) && \text{(adjacent matrix} \rightarrow \mathbf{I} \text{ as } \epsilon \rightarrow 0^+) \\ &= \text{Tr} \left( \lim_{\epsilon \rightarrow 0^+} \frac{\partial}{\partial \epsilon} \frac{\partial}{\partial \mathbf{z}(t)} (\mathbf{z}(t) + \epsilon \mathbf{f}(\mathbf{z}(t), t) + o(\epsilon^2) + \dots) \right) \\ &= \text{Tr} \left( \lim_{\epsilon \rightarrow 0^+} \frac{\partial}{\partial \epsilon} \left( \mathbf{I} + \epsilon \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} + o(\epsilon^2) + \dots \right) \right) \\ &= \text{Tr} \left( \lim_{\epsilon \rightarrow 0^+} \left( \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} + o(\epsilon) + \dots \right) \right) \\ &= \nabla \cdot \mathbf{f}(\mathbf{z}(t), t). \end{aligned}$$

Before deriving the second term, we take the first-order Taylor expansion of  $\log \mu(\Phi_\epsilon(\mathbf{z}(t)))$  at  $\mathbf{z}(t) = \Phi_t(\mathbf{x})$ :

$\log \mu(\Phi_{t+\epsilon}(\mathbf{x})) = \log \mu(\Phi_\epsilon(\mathbf{z}(t))) = \log \mu(\mathbf{z}(t)) + \nabla \log \mu(\mathbf{z}(t)) \cdot (\Phi_\epsilon(\mathbf{z}(t)) - \mathbf{z}(t)) + o(\epsilon^2) + \dots$ ,  
hence,

$$\begin{aligned} & \lim_{\epsilon \rightarrow 0^+} \frac{\log \mu(\Phi_{t+\epsilon}(\mathbf{x})) - \log \mu(\Phi_t(\mathbf{x}))}{\epsilon} \\ &= \lim_{\epsilon \rightarrow 0^+} \frac{\log \mu(\mathbf{z}(t)) + \nabla \log \mu(\mathbf{z}(t)) \cdot (\Phi_\epsilon(\mathbf{z}(t)) - \mathbf{z}(t)) + o(\epsilon^2) + \dots - \log \mu(\mathbf{z}(t))}{\epsilon} \\ &= \lim_{\epsilon \rightarrow 0^+} \nabla \log \mu(\mathbf{z}(t)) \cdot \frac{\Phi_\epsilon(\mathbf{z}(t)) - \mathbf{z}(t)}{\epsilon} + o(\epsilon) + \dots \\ &= \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t). \end{aligned}$$

Therefore, the differential of  $\log \tilde{p}_t(\mathbf{x})$  is:

$$\frac{d \log \tilde{p}_t(\mathbf{x})}{dt} = \nabla \cdot \mathbf{f}(\mathbf{z}(t), t) + \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t).$$

□

To show the relation between two differentials  $\frac{d \log \tilde{p}_t(\mathbf{x})}{dt}$  and  $\frac{d \log p_t(\mathbf{z}(t))}{dt}$ , we first need the relation between  $\log \tilde{p}_t(\mathbf{x})$  and  $\log p_t(\mathbf{z}(t))$ :

$$\log \tilde{p}_t(\mathbf{x}) = \log p(\mathbf{x}) + \log \mu(\mathbf{z}(t)) - \log p_t(\mathbf{z}(t)).$$

Taking the total derivative on both l.h.s. and r.h.s. of last equation:

$$\begin{aligned} \frac{d \log \tilde{p}_t(\mathbf{x})}{dt} &= \frac{d \log \mu(\mathbf{z}(t))}{dt} - \frac{d \log p_t(\mathbf{z}(t))}{dt} \\ &= \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t) - \frac{d \log p_t(\mathbf{z}(t))}{dt} \\ &= \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t) + \nabla \cdot \mathbf{f}(\mathbf{z}(t), t). \end{aligned}$$

The total derivative  $\frac{d \log \tilde{p}_t(\mathbf{x})}{dt}$  is defined on the fixed variable  $\mathbf{x}$ , while the infinitesimal change on r.h.s. is evaluated on the variable  $\mathbf{z}(t)$ . So solving  $\log \tilde{p}_t(\mathbf{x})$  requires to simulate  $\mathbf{z}(t)$  simultaneously. Different to solving  $\log p_t(\mathbf{z}(t))$  on the reversed direction of solving  $\mathbf{z}(t)$ ,  $\log \tilde{p}_t(\mathbf{x})$  only needs the trajectory of  $\mathbf{z}(\tau)$ ,  $\tau \in [0, t]$ , while  $\log p_t(\mathbf{z}(t))$  requires to know the whole trajectory of  $\mathbf{z}(\tau)$ ,  $\tau \in [0, T]$ . Therefore, using  $\log \tilde{p}_t(\mathbf{x})$  is more advantageous when evaluating models at any  $t$  other than  $T$  or at multiple  $t$ .

As for training, since  $p_T$  is specified as  $\mu$  at  $T$ , maximizing  $\log p(\mathbf{x})$  in vanilla CNF is essentially equivalent to maximizing  $\log \tilde{p}_t(\mathbf{x})$  in ACNF.

If we take the time partial derivative on the log-likelihood equation, then

$$\frac{\partial \log \tilde{p}_t(\mathbf{x})}{\partial t} = \frac{\partial \log \mu(\mathbf{z}(t))}{\partial t} - \frac{\partial \log p_t(\mathbf{z}(t))}{\partial t} = -\frac{\partial \log p_t(\mathbf{z}(t))}{\partial t},$$

so that the convergence rate of distribution estimate  $\tilde{p}_t(\mathbf{x})$  towards  $p(\mathbf{x})$  is equivalent to the normalized distribution  $p_t(\mathbf{z})$  towards  $\mu(\mathbf{z})$ .

## A.2 PROOF OF THE DYNAMICS FOR THE STEEPEST ASCENT CONTINUOUS NORMALIZING FLOWS

**Theorem** (Dynamics for Steepest Ascent Continuous Normalizing Flows). *Let  $\mathbf{z}(t)$  be a finite continuous random variable and the solution of a differential equation  $\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}(\mathbf{z}(t), t)$  with initial*

value  $\mathbf{z}(0) = \mathbf{x}$ . Its probability  $p_t(\mathbf{z}(t))$  subjects to the continuity equation  $\partial_t p_t + \nabla \cdot (p_t \mathbf{f}) = 0$ . The dynamics of the steepest flow for decreasing  $KL(p_t(\mathbf{z}(t)) || \mu(\mathbf{z}(t)))$  is

$$\mathbf{f}^*(\mathbf{z}(t), t) = \nabla \log \mu(\mathbf{z}(t)) - \frac{\nabla p_t(\mathbf{z}(t))}{p_t(\mathbf{z}(t))} = \nabla \log \mu(\mathbf{z}(t)) - \nabla \log p_t(\mathbf{z}(t)).$$

To keep this proof simple, we derive this theorem in Euclidean space. If readers are familiar with non-Euclidean metric spaces, we refer more rigid of Wasserstein gradient flow proof in (Ambrosio et al., 2005).

*Proof.* Assuming that  $N$  samples  $X = \{\mathbf{x}_i\}_{i=1:N} \in \mathbb{R}^{Nd}$  are drawn from  $p(\mathbf{x})$ , the averaged negative estimated log-likelihood at time  $t$  is:

$$J(\Phi_t) = -\frac{1}{N} \sum_{i=1}^N \log \tilde{p}_t(\mathbf{x}_i) = \frac{1}{N} \sum_{i=1}^N (\log p_t(\Phi_t(\mathbf{x}_i)) - \log \mu(\Phi_t(\mathbf{x}_i)) - \log p(\mathbf{x}_i)).$$

Using the chain rule, the derivative of  $J(\Phi_t)$  w.r.t.  $\Phi_t(\mathbf{x}_i)$  is:

$$[\nabla J(\Phi_t)]_i = \nabla \log p_t(\Phi_t(\mathbf{x}_i)) - \nabla \log \mu(\Phi_t(\mathbf{x}_i)),$$

where  $\nabla J(\Phi_t)$  is a matrix that each row is for each sample  $i = 1, 2, \dots, N$  and each column is for each dimension  $j = 1, 2, \dots, d$ .

To numerically compute the solutions of Euler-Lagrange equation, i.e.  $\nabla J(\Phi_t) = \mathbf{0}$ , we use *gradient descent* to define the evolution of transformation  $\Phi_t$  for each  $\mathbf{x}_i$ :

$$\frac{d\Phi_t(\mathbf{x}_i)}{dt} = -[\nabla J(\Phi_t)]_i = \nabla \log \mu(\Phi_t(\mathbf{x}_i)) - \nabla \log p_t(\Phi_t(\mathbf{x}_i)),$$

which evolves  $\Phi$  in the direction that decreases  $J(\Phi)$  most rapidly, starting at initial  $\Phi_0(\mathbf{x}_i) = \mathbf{x}_i$ .

The next step is to extend the assumption of the finite number of data samples  $N$  to infinity, i.e.  $N \rightarrow \infty$ , therefore, the objective  $J(\Phi_t)$  at time  $t$  is updated as:

$$\begin{aligned} J(\Phi_t) &= - \int_U \log \tilde{p}_t(\mathbf{x}) d\mathbf{x} \\ &= \int_U (\log p_t(\Phi_t(\mathbf{x})) - \log \mu(\Phi_t(\mathbf{x})) - \log p(\mathbf{x})) d\mathbf{x} \\ &= \int_U L(\mathbf{x}, \Phi_t(\mathbf{x}), \nabla \Phi_t(\mathbf{x})) d\mathbf{x}, \end{aligned}$$

where  $\mathbf{x} \in U \subseteq \mathbb{R}^d$  and  $L(\mathbf{x}, \Phi_t(\mathbf{x}), \nabla \Phi_t(\mathbf{x})) = \log p_t(\Phi_t(\mathbf{x})) - \log \mu(\Phi_t(\mathbf{x})) - \log p(\mathbf{x})$ . For each  $j$  dimension of  $\Phi_t$ , the functional derivative of  $J(\Phi_t)$  w.r.t.  $[\Phi_t]_j$  is:

$$\begin{aligned} \frac{\delta J(\Phi_t)}{\delta [\Phi_t]_j} &= \frac{\partial L}{\partial [\Phi_t]_j}(\mathbf{x}, \Phi_t(\mathbf{x}), \nabla \Phi_t(\mathbf{x})) - \nabla \cdot \left( \frac{\partial L}{\partial \nabla [\Phi_t]_j}(\mathbf{x}, \Phi_t(\mathbf{x}), \nabla \Phi_t(\mathbf{x})) \right) \\ &= [\nabla \log p_t(\Phi_t(\mathbf{x}))]_j - [\nabla \log \mu(\Phi_t(\mathbf{x}))]_j, \end{aligned}$$

as  $\frac{\partial L}{\partial \nabla [\Phi_t]_j} = \mathbf{0}$ . Therefore, the gradient descent that defines the evolution of transformation  $\Phi_t$  is:

$$\frac{d\Phi_t(\mathbf{x})}{dt} = - \frac{\delta J(\Phi_t)}{\delta \Phi_t} = \nabla \log \mu(\Phi_t(\mathbf{x})) - \nabla \log p_t(\Phi_t(\mathbf{x})), \quad (11)$$

therefore, the dynamics for the steepest ascent continuous normalizing flow is:

$$\mathbf{f}^*(\mathbf{z}(t), t) = \frac{d\Phi_t(\mathbf{x})}{dt} = \nabla \log \mu(\mathbf{z}(t)) - \nabla \log p_t(\mathbf{z}(t)).$$

□

### A.3 CONVERGENCE RATE OF OPTIMAL ASCENT CONTINUOUS NORMALIZING FLOWS AND ITS RELATION TO LANGEVIN DYNAMICS

The convergence rate of KL divergence w.r.t.  $t$  can be derived as (we start from a general flow dynamics  $\mathbf{f}$ ):

$$\begin{aligned}
\frac{\partial}{\partial t} \text{KL}(p_t(\mathbf{z}) \| \mu(\mathbf{z})) &= \frac{\partial}{\partial t} \text{KL}(p(\mathbf{x}) \| \tilde{p}_t(\mathbf{x})) \\
&= - \int p(\mathbf{x}) \frac{\partial}{\partial t} \tilde{p}_t(\mathbf{x}) d\mathbf{x} \\
&= - \int p(\mathbf{x}) (\nabla \cdot \mathbf{f}(\mathbf{z}(t), t) + \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t)) d\mathbf{x} \\
&= - \int p_t(\mathbf{z}(t)) (\nabla \cdot \mathbf{f}(\mathbf{z}(t), t) + \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t)) d\mathbf{z}(t) \\
&= - \int p_t(\mathbf{z}) \left( \sum_i \frac{\partial f_i(\mathbf{z}, t)}{\partial z_i} + \sum_i \frac{\partial \log \mu(\mathbf{z})}{\partial z_i} f_i(\mathbf{z}) \right) d\mathbf{z} \tag{12} \\
&= - \sum_i \left[ -f_i(\mathbf{z}, t) \frac{\partial p_t(\mathbf{z})}{\partial z_i} + \frac{\partial \log \mu(\mathbf{z})}{\partial z_i} f_i(\mathbf{z}) \right] \\
&= - \sum_i \int p_t(\mathbf{z}) \left( -\frac{\partial \log p_t(\mathbf{z})}{\partial z_i} + \frac{\partial \log \mu(\mathbf{z})}{\partial z_i} \right) f_i(\mathbf{z}, t) d\mathbf{z} \\
&= -\mathbb{E}_{p_t} [(\nabla \log \mu(\mathbf{z}) - \nabla \log p_t(\mathbf{z})) \cdot \mathbf{f}(\mathbf{z}, t)].
\end{aligned}$$

(Liu 2017)[theorem 3.1] shows similar derivation from discrete transformation perspective and links to Stein variational gradient flows.

When dynamics  $\mathbf{f}$  is equal to the fastest flow dynamics  $\mathbf{f}^*$  as eq.(6), then the convergence rate becomes negative Fisher divergence (estimated w.r.t.  $p_t$ ):

$$\frac{\partial}{\partial t} \text{KL}(p_t(\mathbf{z}) \| \mu(\mathbf{z})) = -\mathbb{E}_{p_t} \|\nabla \log p_t(\mathbf{z}) - \nabla \log \mu(\mathbf{z})\|_2^2.$$

This convergence rate can be easily proved the same to overdamped Langevin diffusion dynamics which is defined via a stochastic differential equation at case of  $\beta = 1$ :

$$d\mathbf{z}(t) = \nabla \log \mu(\mathbf{z}(t)) dt + \sqrt{2\beta^{-1}} d\mathbf{W}_t, \tag{13}$$

where  $\mathbf{W}_t$  is a Brownian motion. Under the Langevin dynamics, the transformed distribution has a PDE:

$$\begin{aligned}
\frac{\partial p_t(\mathbf{z})}{\partial t} &= -\nabla \cdot (p_t(\mathbf{z}) \nabla \log \mu(\mathbf{z})) + \beta^{-1} \Delta p_t(\mathbf{z}) \\
&= -\nabla \cdot (p_t(\mathbf{z}) \nabla \log \mu(\mathbf{z})) + \beta^{-1} \nabla \cdot (\nabla p_t(\mathbf{z})) \\
&= -\nabla \cdot (p_t(\mathbf{z}) (\nabla \log \mu(\mathbf{z}) - \beta^{-1} \nabla \log p_t(\mathbf{z}))).
\end{aligned}$$

The last line reveals the steepest gradient flow dynamics as eq.(6) when  $\beta = 1$ .

Therefore, the optimal ascent continuous normalizing flows and overdamped Langevin dynamics transform a distribution equivalently when  $\beta = 1$ . And this Fokker Plank equation is a linear (w.r.t.  $p_t(\mathbf{z})$ ) and deterministic although Langevin dynamics is stochastic. The main difference between these two flows is that the dynamics of (optimal) ascent continuous normalizing flow is deterministic, so as any particular sample trajectory; while Langevin dynamics defines a stochastic process and sample trajectories are stochastic.

### A.4 DERIVATION OF POTENTIAL FIELD PDE

The optimal dynamics defined in eq.(6) can be rewritten in terms of the potential function  $V(\mathbf{z}(t), t)$ , as  $V(\mathbf{z}, t) := \frac{p_t(\mathbf{z})}{\mu(\mathbf{z})}$ :

$$\mathbf{f}^* = \nabla \log \mu(\mathbf{z}(t)) - \nabla \log p(\mathbf{z}(t), t) = -\nabla \log V(\mathbf{z}(t), t). \tag{14}$$



The continuity equation reveals the time derivative of the transformed density  $p(\mathbf{z}(t), t)$  at  $t$ :

$$\begin{aligned}\frac{\partial p_t(\mathbf{z}(t))}{\partial t} &= -\nabla \cdot (p_t(\mathbf{z}(t))\mathbf{f}(\mathbf{z}(t), t)) \\ &= -p_t(\mathbf{z}(t))\nabla \cdot \mathbf{f}(\mathbf{z}(t), t) - \nabla p_t(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t).\end{aligned}$$

Therefore, the time derivative of  $\log p_t(\mathbf{z})$  with dynamics defined in eq. (14) is:

$$\begin{aligned}\frac{\partial \log p_t(\mathbf{z}(t))}{\partial t} &= \frac{1}{p_t(\mathbf{z}(t))} \frac{\partial p_t(\mathbf{z}(t))}{\partial t} = -\nabla \cdot \mathbf{f}(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t) \\ &= \Delta \log V(\mathbf{z}(t), t) + \nabla \log p_t(\mathbf{z}(t)) \cdot \nabla \log V(\mathbf{z}(t), t).\end{aligned}$$

Using the last equation, the time derivative of  $\log V(\mathbf{z}, t)$  is derived as:

$$\begin{aligned}\frac{\partial \log V(\mathbf{z}, t)}{\partial t} &:= \frac{\partial \log p_t(\mathbf{z}(t))}{\partial t} - \frac{\partial \log \mu(\mathbf{z}(t))}{\partial t} \\ &= -\nabla \cdot \mathbf{f}(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t) - \nabla \log \mu(\mathbf{z}(t)) \cdot \mathbf{f}(\mathbf{z}(t), t) \\ &= \Delta \log V(\mathbf{z}(t), t) + (\nabla \log p_t(\mathbf{z}(t)) + \nabla \log \mu(\mathbf{z}(t))) \cdot \nabla \log V(\mathbf{z}(t), t) \\ &= \Delta \log V(\mathbf{z}(t), t) + (2\nabla \log \mu(\mathbf{z}(t)) + \nabla \log V(\mathbf{z}(t), t)) \cdot \nabla \log V(\mathbf{z}(t), t),\end{aligned}$$

therefore, the time derivative of potential field is:

$$\frac{\partial V(\mathbf{z}, t)}{\partial t} = \Delta V(\mathbf{z}, t) + 2\nabla \log \mu(\mathbf{z}) \cdot \nabla V(\mathbf{z}, t) + \nabla \log V(\mathbf{z}, t) \cdot \nabla V(\mathbf{z}, t). \quad (15)$$

When  $t = 0$ ,  $V(\mathbf{x}, 0) = \frac{p(\mathbf{x})}{\mu(\mathbf{x})}$ ; when  $t \rightarrow \infty$ ,  $V(\mathbf{z}, t) \equiv 1, \forall \mathbf{z}$ .

#### A.5 INSTANTANEOUS CHANGE OF SCORE FUNCTION

**Theorem** (Instantaneous Change of Score Function). *Let  $\mathbf{z}(t)$  be a finite continuous random variable with probability density  $p_t(\mathbf{z}(t))$  at time  $t$ . Let  $\frac{d\mathbf{z}(t)}{dt} = \mathbf{f}(\mathbf{z}(t), t)$  be a differential equation describing a continuous-in-time transformation of  $\mathbf{z}(t)$ . Assuming that  $\mathbf{f}$  is uniformly Lipschitz continuous in  $\mathbf{z}$  and  $t$ , the infinitesimal change in the gradient of log-density at  $t$  is*

$$\frac{d\nabla \log p_t(\mathbf{z}(t))}{dt} = -\nabla \log p_t(\mathbf{z}(t)) \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} - \nabla (\nabla \cdot \mathbf{f}(\mathbf{z}(t), t)).$$

*Proof.* As  $\mathbf{f}$  is assumed to be Lipschitz continuous in  $\mathbf{z}(t)$  and  $t$ ,  $\Phi_t(\mathbf{x})$  represents the unique solution map. We denote the transformation on  $\mathbf{z}(t + \epsilon)$  reversed over an  $\epsilon$  change in time as:

$$\mathbf{z}(t + \epsilon) = \Phi_\epsilon(\mathbf{z}(t)), \quad \mathbf{z}(t) = \Phi_{-\epsilon}(\mathbf{z}(t + \epsilon)),$$

and applying the change of variable theorem on  $\log p_{t+\epsilon}(\mathbf{z}(t + \epsilon))$ , defined on the variable  $\mathbf{z}(t + \epsilon)$ :

$$\begin{aligned}\log p_{t+\epsilon}(\mathbf{z}(t + \epsilon)) &= \log p_t(\mathbf{z}(t)) - \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))| \\ &= \log p_t(\Phi_{-\epsilon}(\mathbf{z}(t + \epsilon))) - \log |\det \mathcal{J}_{\Phi_\epsilon}(\Phi_{-\epsilon}(\mathbf{z}(t + \epsilon)))|.\end{aligned}$$

Taking the derivative of  $\log p_{t+\epsilon}(\mathbf{z}(t + \epsilon))$  w.r.t.  $\mathbf{z}(t + \epsilon)$  on both l.h.s. and r.h.s. of the last equation:

$$\nabla \log p_{t+\epsilon}(\mathbf{z}(t + \epsilon)) = (\nabla \log p_t(\mathbf{z}(t)) - \nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|) \frac{\partial \Phi_{-\epsilon}(\mathbf{z}(t + \epsilon))}{\partial \mathbf{z}(t + \epsilon)},$$

and the infinitesimal limit of finite changes of gradient of log density can be defined:

$$\begin{aligned}
\frac{d\nabla \log p_t(\mathbf{z}(t))}{dt} &:= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} (\nabla \log p_{t+\epsilon}(\mathbf{z}(t+\epsilon)) - \nabla \log p_t(\mathbf{z}(t))) \\
&= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( (\nabla \log p_t(\mathbf{z}(t)) - \nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|) \frac{\partial \Phi_{-\epsilon}(\mathbf{z}(t+\epsilon))}{\partial \mathbf{z}(t+\epsilon)} - \nabla \log p_t(\mathbf{z}(t)) \right) \\
&= \nabla \log p_t(\mathbf{z}(t)) \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \left( \frac{\partial \Phi_\epsilon(\mathbf{z}(t))}{\partial \mathbf{z}(t)} \right)^{-1} - \mathbf{I} \right) - \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))| \left( \frac{\partial \Phi_\epsilon(\mathbf{z}(t))}{\partial \mathbf{z}(t)} \right)^{-1} \right) \\
&= -\nabla \log p_t(\mathbf{z}(t)) \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} - \nabla (\nabla \cdot \mathbf{f}(\mathbf{z}(t), t)),
\end{aligned} \tag{16}$$

where the two limits are derived in detail:

$$\begin{aligned}
&\lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \left( \frac{\partial \Phi_\epsilon(\mathbf{z}(t))}{\partial \mathbf{z}(t)} \right)^{-1} - \mathbf{I} \right) \\
&= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \left( \frac{\partial}{\partial \mathbf{z}(t)} (\mathbf{z}(t) + \epsilon \mathbf{f}(\mathbf{z}(t), t) + o(\epsilon^2) + \dots) \right)^{-1} - \mathbf{I} \right) \\
&= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \left( \mathbf{I} + \epsilon \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} + o(\epsilon^2) + \dots \right)^{-1} - \mathbf{I} \right) \\
&= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \left( \mathbf{I} - \epsilon \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} + o(\epsilon^2) + \dots \right) - \mathbf{I} \right) \quad (\text{inverse by geometric power series expansion}) \\
&= \lim_{\epsilon \rightarrow 0^+} -\frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} + o(\epsilon) + \dots \\
&= -\frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)},
\end{aligned}$$

and

$$\begin{aligned}
&\lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))| \left( \frac{\partial \Phi_\epsilon(\mathbf{z}(t))}{\partial \mathbf{z}(t)} \right)^{-1} \right) \\
&= \lim_{\epsilon \rightarrow 0^+} \frac{1}{\epsilon} \left( \nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))| \left( \mathbf{I} - \epsilon \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} + o(\epsilon^2) + \dots \right) \right) \\
&= \lim_{\epsilon \rightarrow 0^+} \frac{\nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|}{\epsilon} - \underbrace{\lim_{\epsilon \rightarrow 0^+} \nabla \log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|}_{\nabla 1 \rightarrow 0} \frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)} \\
&= \nabla \lim_{\epsilon \rightarrow 0^+} \frac{\log |\det \mathcal{J}_{\Phi_\epsilon}(\mathbf{z}(t))|}{\epsilon} \\
&= \nabla (\nabla \cdot \mathbf{f}(\mathbf{z}(t), t)).
\end{aligned} \tag{17}$$

□

Therefore,  $\nabla \log p(\mathbf{x}, t)$  follows a linear matrix differential equation, where the linear matrix is defined by the Jacobian  $\frac{\partial \mathbf{f}(\mathbf{z}(t), t)}{\partial \mathbf{z}(t)}$  and the bias term is the gradient of divergence of the differential function  $\nabla (\nabla \cdot \mathbf{f}(\mathbf{z}(t), t))$ .

#### A.6 INTERPRETING ASCENT REGULARIZATION AS SCORE MATCHING OBJECTIVE

To show the ascent regularization in eq.(9) and eq.(10) relates to the score matching objective, we first assume a diffusion process defined via a stochastic differential equation (SDE):

$$d\mathbf{z}(t) = \mathbf{h}(\mathbf{z}(t), t) + g(t)d\mathbf{W}(t), \mathbf{z}(0) = \mathbf{x}; \mathbf{x} \sim p(\mathbf{x}), \tag{18}$$

where  $\mathbf{W}_t$  is Brownian motion and we denote  $p_t(\mathbf{z}(t))$  as the marginal distribution at time  $t$  and  $\mathcal{P}_T$  as the path measure of the SDE up to time  $T$ .

(Anderson, 1982) shows the reverse time process is also a diffusion process which shares the same marginals as the forward process:

$$d\mathbf{z}(t) = (\mathbf{h}(\mathbf{z}(t), t) - g^2(t)\nabla \log p_t(\mathbf{z}(t))) dt + g(t)d\tilde{\mathbf{W}}(t), \mathbf{z}(T) \sim p_T, \quad (19)$$

where  $\tilde{\mathbf{W}}(t)$  is a reverse-time Brownian motion. The reverse-time diffusion introduces the conditional path measure  $\mathcal{P}(\cdot|\mathbf{z}(T))$ . As the score function,  $\nabla \log p_t(\mathbf{z}(t))$ , is generally unknown for an arbitrary diffusion process, we approximate the reverse-time diffusion by a secondary reverse-time diffusion process by a parametric score function:

$$d\mathbf{z}(t) = (\mathbf{h}(\mathbf{z}(t), t) - g^2(t)s_\theta(\mathbf{z}(t), t)) dt + g(t)d\tilde{\mathbf{W}}(t), \mathbf{z}(T) \sim p_T, \quad (20)$$

which induces the conditional path measure  $\tilde{\mathcal{P}}_T^\theta(\cdot|\mathbf{z}(T))$  to approximate  $\mathcal{P}_T(\cdot|\mathbf{z}(T))$ .

Under some regularity conditions that permit the definition of Radon-Nikodym derivative,  $d\mathcal{P}_T(\cdot|\mathbf{z}(T))/d\tilde{\mathcal{P}}_T^\theta(\cdot|\mathbf{z}(T))$ , Girsanov theorem gives the expectation of KL divergence between two path measures:

$$\begin{aligned} \mathbb{E}_{p_T} \left[ \text{KL}(\mathcal{P}_T(\cdot|\mathbf{z}(T)) \parallel \tilde{\mathcal{P}}_T^\theta(\cdot|\mathbf{z}(T))) \right] &= -\mathbb{E}_{\mathcal{P}} \left[ \log \frac{d\tilde{\mathcal{P}}_T^\theta(\cdot|\mathbf{z}(T))}{d\mathcal{P}_T(\cdot|\mathbf{z}(T))} \right] \\ &= \mathbb{E}_{\mathcal{P}} \left[ \int_0^T g(t) (s_\theta(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t))) d\tilde{\mathbf{W}}_t + \frac{1}{2} \int_0^T g^2(t) \|s_\theta(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t))\|^2 dt \right] \\ &= \frac{1}{2} \mathbb{E}_{\mathcal{P}} \left[ g^2(t) \int_0^T \|s_\theta(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t))\|^2 dt \right]. \end{aligned}$$

Using the chain rule of KL divergence, we can show the KL divergence between two path measures:

$$\begin{aligned} \text{KL}(\mathcal{P}_T \parallel \tilde{\mathcal{P}}_T^\theta) &= \text{KL}(p_T(\mathbf{z}(T)) \parallel \mu(\mathbf{z}(T))) + \mathbb{E}_{p_T} \left[ \text{KL}(\mathcal{P}_T(\cdot|\mathbf{z}(T)) \parallel \tilde{\mathcal{P}}_T^\theta(\cdot|\mathbf{z}(T))) \right] \\ &= \text{KL}(p_T(\mathbf{z}(T)) \parallel \mu(\mathbf{z}(T))) + \frac{1}{2} \mathbb{E}_{\mathcal{P}} \left[ g^2(t) \int_0^T \|s_\theta(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t))\|^2 dt \right] \\ &= \text{KL}(p(\mathbf{x}) \parallel \tilde{p}(\mathbf{x})) + \frac{1}{2} \mathbb{E}_{\mathcal{P}} \left[ \int_0^T g^2(t) \|s_\theta(\mathbf{z}(t), t) - \nabla \log p_t(\mathbf{z}(t))\|^2 dt \right]. \end{aligned} \quad (21)$$

Assume that the parametric dynamics  $\mathbf{f}(\mathbf{z}(t), t; \theta) = \nabla \log \mu(\mathbf{z}(t)) - s_\theta(\mathbf{z}(t), t)$  has the similar structure as the optimal dynamics in eq.(6) as  $s_\theta(\mathbf{z}(t), t)$  to approximate  $\nabla \log p_t(\mathbf{z}(t))$  and  $g(t) \equiv \sqrt{2\lambda}$ , then we recover the total learning objective with ascent regularization coefficient  $\lambda$  in eq.(9). Therefore, the total objective is equivalent to minimize the KL divergence of two path measures on the joint (infinite) variable space. Similar analysis can also be applied to the objective in eq.(10).

When  $\lambda = \beta^{-1} = \frac{g^2(t)}{2}$  and learned score  $s_\theta(\mathbf{z}(t), t)$  matches to  $\nabla \log p_t(\mathbf{z}(t))$  so that  $\mathbf{h}(\mathbf{z}(t), t) = \nabla \log \mu(\mathbf{z}(t))$ , then SDE in eq.(18) becomes the overdamped Langevin dynamics in eq.(13) as well as optimal ACNF (eq.(6)) with critical damping dynamics, i.e.  $\lambda = \beta^{-1} = 1$ .

As the ascent regularization can be interpreted as a score matching objective, it is possible to implement Algorithm 1 and Algorithm 2 in a more time efficient way for training like (Lu et al., 2022; Song et al., 2021). However, note that the explicit score matching objective can hardly be used directly in the implementation as  $\nabla \log p_t(\mathbf{z}(t))$  is intractable in general and requires to be evaluated e.g. via score function integral in ascent regularization. (Lu et al., 2022; Song et al., 2021; 2020; Ho et al., 2020) use its surrogates, e.g. denoising score matching. To enable practical training, denoising score matching objective relies on the explicit form of conditional (noised) distributions  $\nabla \log p_{t|0}(\mathbf{z}(t)|\mathbf{z}(0))$ , e.g. Gaussian. For image or data generation tasks, Gaussian assumption may

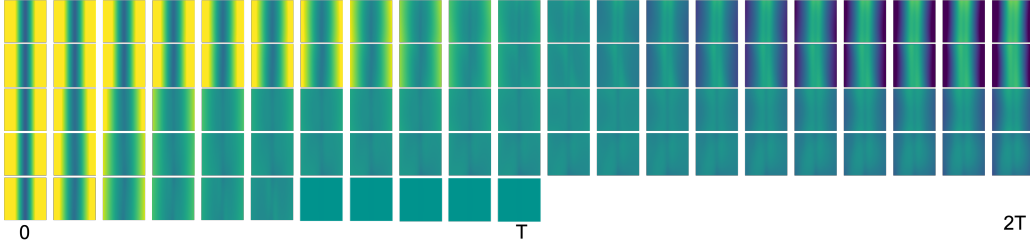


Figure 10: Comparison of log potential field,  $\log V(\mathbf{z}(t), t)$ , evaluated on trained vanilla CNF, RNODE with regularization coefficient as 0.1 and ACNF models with regularization coefficient  $\lambda$  as 0.1 and 1 for 2-modal Gaussian mixture along flow at  $t \in [0, 2T]$  and the numerical PDE solutions of eq.(7). Color indicates the value of field: turquoise is 0, and the lighter the color is the larger the value is, and vice versa.

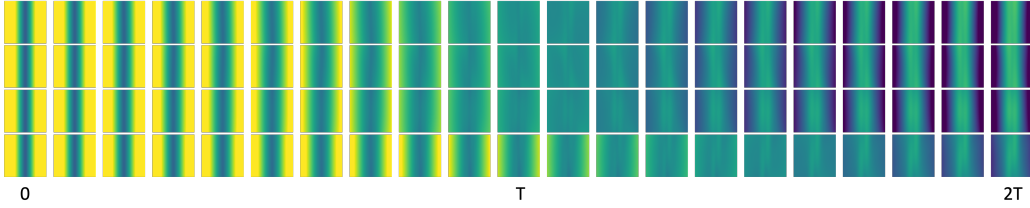


Figure 11: Comparison of log potential field,  $\log V(\mathbf{z}(t), t)$ , evaluated on trained vanilla CNF and RNODE (Finlay et al., 2020) models with  $T = 1$  for 2-modal Gaussian mixture along the flows at  $t \in [0, 2T]$  as Figure 4. The kinetic energy regularization coefficients are 0, 0.01, 0.1, 1 respectively. Color indicates the value of field: turquoise is 0, and the lighter the color is the larger the value is, and vice versa.

not seem so limited as long as the chain of discrete transformation is adequately long for adequate expressivity of the marginal distribution at  $T$ . However, for inference tasks e.g. using flows as variational approximation or annealed sampler, constraining the distribution induced by flows with Gaussian assumption can hinder their approximate potential for true posterior.

#### A.7 ANALYSIS ON A TOY EXAMPLE: FROM A GAUSSIAN TO A MIXTURE OF GAUSSIAN

Before we deploy ACNF for complex distributions, we first demonstrate its validity on a simpler problem: to learn a 2-modal Gaussian mixture with a standard Gaussian as the base distribution. Since the density of the target distribution is known in this case, we can numerically solve the potential field  $V(\mathbf{z}, t)$  for  $t \in [0, T]$  in eq.(7) even though the exact solution is still hard to obtain for this simple case.

The PDE solution presented in Figure 4 and Figure 10 is implemented using `py-pde` package. A fixed Cartesian grid is used which has the same center locations as the other potential fields evaluated by density estimations. The PDE solver in `py-pde` uses the finite difference method, and we choose explicit solver to keep simulation simple.

To define the parametric dynamics function for training, we use hypernetworks (Ha et al., 2016) that a smaller network generates the weights of layers. This architecture is suitable to demonstrate ACNFs as the function of dynamics is supposed to evolve with time via changing the weights by the hypernetworks. We follow the same implementation of hypernetworks as Neural ODE and use `torchdiff` for ODE solution and adjoint method.<sup>1</sup>

The last row of Figure 10 as Figure 4 shows the logarithm of the potential solutions, while the rest show the log potential field of learned flows evaluated by the ratio  $p(\mathbf{x})/\tilde{p}_t(\mathbf{x})$  when training  $T$  is set as 10.

<sup>1</sup><https://github.com/rtqichen/torchdiffeq>

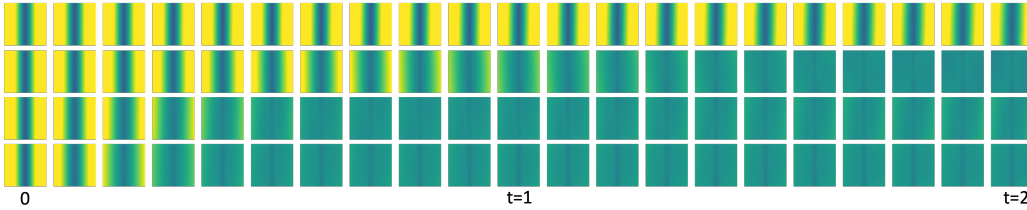


Figure 12: Comparison on log potential field,  $\log V(\mathbf{z}(t), t)$  of trained vanilla CNF and ACNF models with  $T = 5$  for 2-modal Gaussian mixture, evaluated along the flows at  $t \in [0, 2]$  as Figure 4. The ascent regularization coefficients  $\lambda$  are 0, 0.01, 0.1, 1 respectively.

Without ascent regularization, the potential field converges slower and only reaches close to a uniform field at  $T$ . After  $T$ , some areas start to be under/over-represented when the learned flow continues to move samples towards the center of the field. Nevertheless, the flows learned with ascent regularization transform densities faster to the target distribution. When the ascent regularization coefficient  $\lambda$  is 1, the evolution of the potential fields is very similar to that of PDE solutions which indicates the learned flow is close to the optimal ascent continuous normalizing flow.

Apart from vanilla CNF, we train RNODE models to demonstrate the effect of kinetic energy regularization on the transformation of distributions. As known from (Finlay et al., 2020), the optimal flow that minimizes  $L_2$  transport cost induces straight sample trajectories and samples travel with constant speeds. Figure 11 shows the flows of RNODE models trained under the same configurations as Figure 4, and the kinetic energy regularization coefficients are 0, 0.01, 0.1, 1 respectively.

Although RNODEs learn simpler ODE functions with lower NFEs compared to the flow without regularization, these flows do not induce the transformed distributions to converge faster. They are even slower at larger regularization coefficients. Like vanilla CNF, RNODE does not prevent the distribution to deteriorate after  $T$ . NFEs for each flow in Figure 11 at the time that the transformed distribution gives the maximum estimated log-likelihood, are 38, 38, 36, 32, while the flows by ACNF are 26, 32, 36 under  $\lambda = 0.01, 0.1, 1$ , nevertheless ascent regularization does not explicitly regularize for simpler ODE functions. We also tried Frobenius norm regularization on the Jacobian as suggested by (Finlay et al., 2020), HJB regularization (Onken et al., 2021; Yang and Karniadakis, 2020), second-order regularization (Kelly et al., 2020), however, the evolution of potential fields under these regularizations does not differ much to that of vanilla CNF and RNODEs as shown.

To demonstrate the effect of the length of flow  $T$  in training configuration, we train vanilla CNF and ACNFs with other flow length, e.g.  $T = 5$  and ascent regularization factors as 0, 0.01, 0.1, 1, and evaluate the learned flows at  $t \in [0, 2]$  as Figure 4. Under some suitable condition that there exists an optimal ACNF between the base and the target distributions, the flow is almost independent to the choice of flow length  $T$ . Comparing Figure 12 with  $T = 5$  to Figure 4 with  $T = 1$  but testing both on  $t \in [0, 2]$ , the flow by vanilla CNF is idle at early stage for  $T = 5$  and is very sensitive to the choice of  $T$ , while the flows with ascent regularization are almost independent to the choice of  $T$ , which possibly makes tedious model selection on different  $T$  or optimizing  $T$  (Ghosh et al., 2020; Du et al., 2022) no longer necessary.

#### A.8 DENSITY ESTIMATION ON 2D TOY DISTRIBUTIONS

Like Section A.7 we specify dynamics model by hypernetworks and all hypernetworks are defined by one hidden layer with 32 units and 64 for the width of hypernetworks to learn all 2-dimensional distributions.

As shown in the last section, the flows learned with ascent regularization are almost insensitive to  $T$  for Gaussian mixture. To examine whether this conclusion still applies to more complex distributions, we retrain ACNF models with ascent regularization coefficients  $\lambda = 0.0001, 0.0005, 0.001, 0.005$  under different flow lengths  $T = 10, 5, 1, 0.5$ . Figure 13 ( $T = 5$ ) and Figure 14 ( $T = 1$ ) shows the evolution of the density estimations for each model at  $t \in [0, 2T]$  like Figure 5 ( $T = 10$ ). When decreasing  $T$  from 10 to 5, the density estimations are almost identical under the same regularization coefficients. When  $T$  decreases from 5 to 1, the highlighted area shrinks slightly at low regularization coefficients, e.g. 0.0001, 0.0005. Model trained with a smaller  $T$  may require a larger  $\lambda$  to have

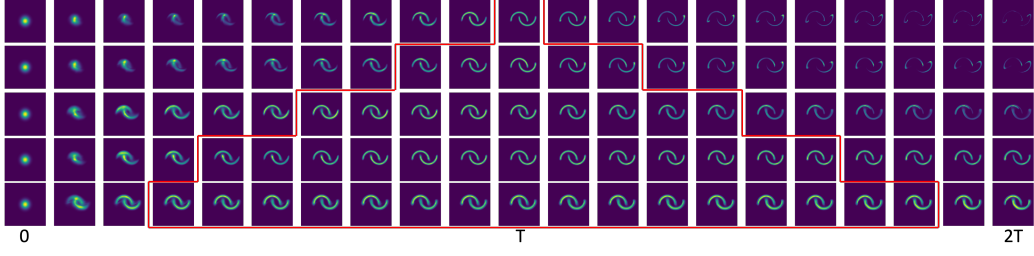


Figure 13: Comparison on density estimations of trained vanilla CNF and ACNFs with regularization coefficients  $\lambda = 0.0001, 0.0005, 0.001, 0.005$  and  $T = 5$  on 2-moon distribution at  $t \in [0, 2T]$ .

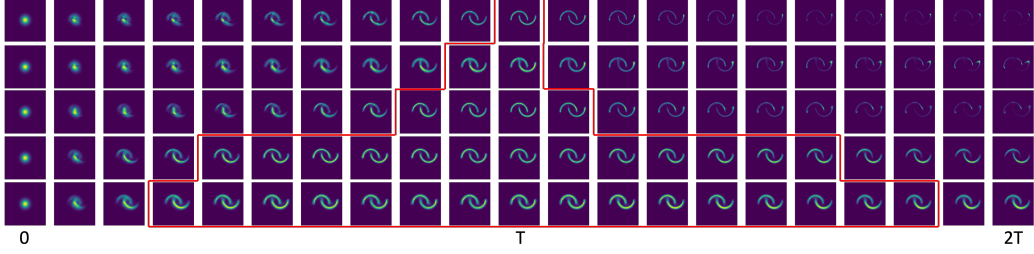


Figure 14: Comparison on density estimations of trained vanilla CNF and ACNFs with regularization coefficients  $\lambda = 0.0001, 0.0005, 0.001, 0.005$  and  $T = 1$ , on 2-moon distribution at  $t \in [0, 2T]$ .

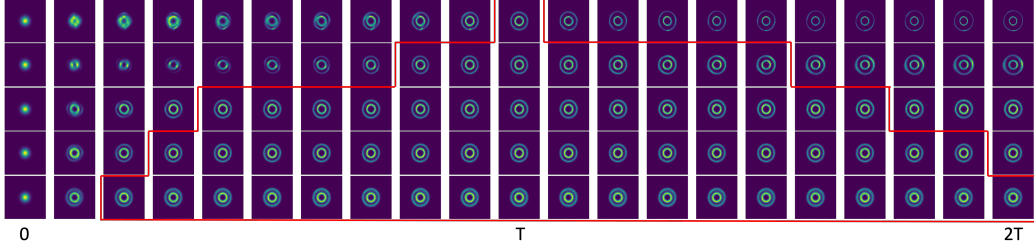


Figure 15: Comparison on density estimations of trained vanilla CNF and ACNFs with regularization coefficients  $\lambda = 0.0001, 0.0005, 0.001, 0.005$  and  $T = 10$  on 2-circle distribution at  $t \in [0, 2T]$ .

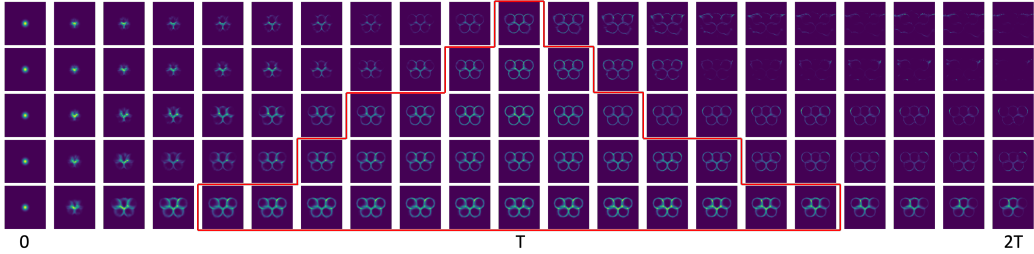


Figure 16: Comparison on density estimations of trained vanilla CNF and ACNFs with regularization coefficients  $\lambda = 0.0001, 0.0005, 0.001, 0.005$  and  $T = 10$  on Olympics distribution at  $t \in [0, 2T]$ .

similar regularization that with a larger  $T$  as the flow length serves as an implicit regularization factor. Although the effect of regularization depends slightly more on the choice of  $T$  for complicated distributions, the flows by ACNF are still much less sensitive to  $T$ , compared to that by CNF.

Apart from the 2-moon distribution, we show the density estimations of learned vanilla CNF and ACNF with different regularization coefficients for modeling 2-circle, Olympics and checkerboard distributions in Figure 15, Figure 16 and Figure 17. They show that ascent regularization is effective in learning different distributions that a larger coefficient induces densities to converge faster to the target distributions and prevents them from deterioration. Comparing across different distributions,

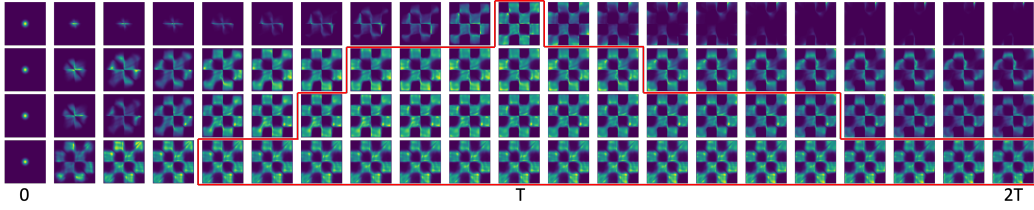


Figure 17: Comparison on density estimations of trained vanilla CNF and ACNFs with regularization coefficients  $\lambda = 0.0001, 0.0005, 0.001$  and  $T = 10$  on checkerboard distribution at  $t \in [0, 2T]$ .

Dataset	# hypernetworks layers	encoding dim	$T$	# flow steps	batch size
POWER	2	6	1	5	10000
GAS	3	4	5	5	1000
HEPMASS	3	10	1	10	10000
MINIBOONE	4	10	1	1	1000
BSDS300	4	10	5	2	10000

Table 3: Model architectures of ACNFs for density estimations on tabular data reported in Table 1.

the highlighted areas are larger for 2-moon, 2-circle and checkerboard distribution than Olympics distributions, since the Olympics distribution is more challenging and requires a relatively large regularization coefficient.

#### A.9 DENSITY ESTIMATION ON TABULAR DATASETS

For tabular datasets, we follow the experiment setup and model configurations as recommended by FFJORD (Grathwohl et al., 2018) and all data are pre-processed according to (Papamakarios et al., 2017). We found that the concatenate layer used in FFJORD, that concatenates time  $t$  and states  $\mathbf{z}(t)$  as a flat input vector for differential function, dilutes the ascent regularization on the parameters, especially when data dimensions are high, e.g. for MINOBOONE and BSDS300 datasets. Nevertheless, the hypernetwork architecture used in previous sections, even a deeper one, turns out to be inadequate to reach a similar log-likelihood evaluation as FFJORD and slow to train. To tackle this issue, we use an encoder to encode states  $\mathbf{z}(t)$  to a lower dimension and apply the weights by the hypernetworks on the encodings and later a decoder maps the transformed encodings back to the data dimension. We summary model architectures and training configurations for each dataset in Table 3.

#### A.10 ACNFs AS ANNEALED SAMPLER FOR UNBIASED SAMPLING AND ESTIMATE OF NORMALIZATION CONSTANT

To extend ACNF annealed sampler with stochasticity, we replace the discrete NF blocks in SNF by the discrete realization of each adaptive step of ACNF and each is followed with a stochastic block by e.g. discrete Langevin flow or MCMC flow as in SNF. The original importance weight update for discrete flows also needs to be replaced by the integral of negative divergence of dynamics and resampling steps are added as AFT (Arbel et al., 2021). The complete algorithm is summarized in Algorithm 3.

Figure 18 shows the generated samples of all different methods as reported in Figure 8 plus adding MC steps on top of trained ACNF to form SNF models by Algorithm 3. Like quantitative evaluation shown in Figure 8, learned ACNFs with regularization coefficient  $\lambda = 0.01$  has distinctly faster convergence than CNF, best tuned linear annealed target and less regularized ACNFs, but uses less computation. The add-on MC steps on trained ACNF boasts the convergence slightly as shown by the last two rows. Although diffeomorphism constraint does not show much effect on limiting the expressiveness of CNF/ACNF in this experiment, adding stochastic blocks is still very beneficial especially at the beginning stage of the flows.



**Algorithm 3** Asymptotically unbiased sampler with learned ACNF  $\mathbf{f}_\theta$ 


---

**Require:** parametric dynamics of ACNF generation flow  $\mathbf{f}_\theta$ , base distribution  $\mu$ , target distribution up to the normalization constant  $\pi(\cdot) = \gamma(\cdot)/Z$ , length of flow  $T$ , number of samples  $N$ , MC step size  $\epsilon$ , number of MC steps  $J$   
sample  $N$  samples from base distribution  $\{\mathbf{z}_0^i\}_{i=1:N} \sim \mu = q_0$   
set  $\log w_0^i = -\log \mu(\mathbf{z}_0^i)$ ,  $t_0 = 0$   
**while**  $t_k < T$  **do**  
    ODE solver chooses step size  $\Delta t_k$ , if  $t_k = t_{k-1} + \Delta t_k < T$  else  $t_k = T$   
    Integrate augmented states  $[\mathbf{z}^i(t), \log q_t(\mathbf{z}^i(t))]$  using generation dynamics  $\mathbf{f}_\theta$  until  $t_k$  from the initial  $[\mathbf{z}_{k-1}^i, \log q_{t_{k-1}}(\mathbf{z}_{k-1}^i)]$  at  $t_{k-1}$   
     $\Delta S_{k,f}^i = \log q_{t_{k-1}}(\mathbf{z}_{k-1}^i) - \log q_{t_k}(\mathbf{z}^i(t_k))$   
     $\mathbf{z}_k^i = \mathbf{z}^i(t_k)$   
    MCMC update with  $\pi$  invariant kernel via Metropolis-Hastings:  
    **for**  $j = 1, \dots, J$  **do**  
        propose  $\mathbf{z}'_k = \mathbf{z}_k + \epsilon \eta^i$ ,  $\eta^i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ,  $\forall i$   
         $a^i = \gamma(\mathbf{z}'_k) / \gamma(\mathbf{z}_k)$ ,  $\forall i$   
        **if**  $\xi^i < a^i$ ,  $\xi^i \sim U(0, 1)$  **then**  
            update  $\mathbf{z}_k = \mathbf{z}'_k$   
        **end if**  
    **end for**  
     $\Delta S_{k,s}^i = \log \gamma(\mathbf{z}^i(t_k)) - \log \gamma(\mathbf{z}_k^i)$ ,  $\forall i$   
    Update weights  $\log w_k^i = \log w_{k-1}^i + \Delta S_{k,f}^i + \Delta S_{k,s}^i$   
    Resample  $\mathbf{z}_k^i$  according to normalized weights  $\tilde{w}_k^i = w_k^i / (\sum_i w_k^i)$   
    Update weights  $w_k^i = 1/N$   
**end while**

---



Figure 18: Comparisons of generated samples from different methods on 2D Gaussian mixture distribution with 8 components as Figure 8. From top to bottom, samples are from: (1-3) linear annealing importance sampler with  $\{170, 25, 10\}$  MC steps between each annealing target; (4) CNF; (5-7) ACNF with ascent regularization factor  $\lambda = 0.0001, 0.001, 0.01$ , (8-9) SNF with trained ACNF  $\lambda = 0.01$  (as 7th row) and  $\{1, 5\}$  MC step as the stochastic block as Algorithm 3.

#### A.11 VARIATIONAL INFERENCE WITH ACNFs

Our experiment setup mimics (Grathwohl et al., 2018), and the encoder and decoder are defined by 7-layer neural networks with specified latent dimension as 64. The first 6 layers of the encoder are implemented as gated convolutional networks and the last one is a linear layer to output mean and diagonal covariance. For the decoder, the first 6 layers are also gated convolutional networks while the last layer is a vanilla convolutional network. We define the length of flow for both VAE-FFJORD and VAE-ACNF as  $T = 1$  and the number of steps as 2. The networks for modeling differential function of flows are the modified hypernetworks as for the tabular datasets, with 4 layers, and the



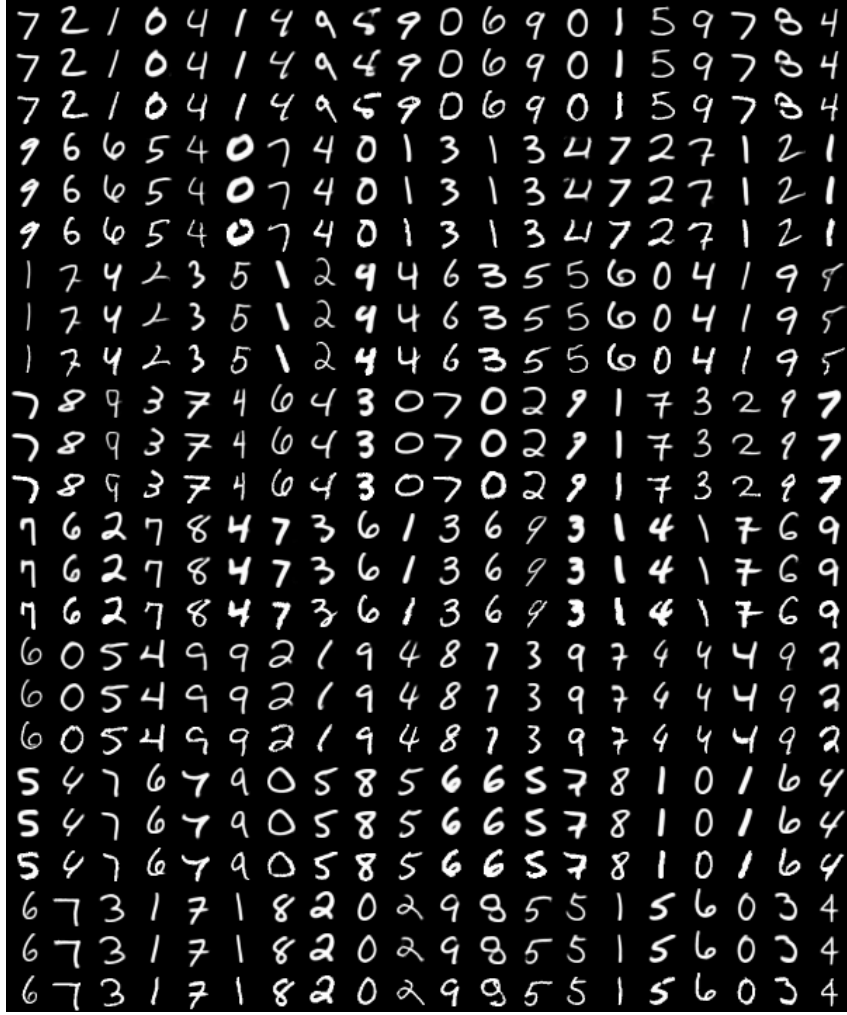


Figure 19: More reconstructed samples from VAE-ACNF, vanilla VAE and original data. The first row of three is the reconstruction from VAE-ACNF, the second one is the reconstruction from vanilla VAE while the last one is the original data samples.

activation function is tanh. All models reported in Table 2 are trained under the same learning rate as 0.001, Adam optimizer and batch size as 100.

Figure 19 shows more reconstructed samples from VAE-ACNF and vanilla VAE, with comparison of original data. In general, the reconstructions from VAE-ACNF are smoother than the ones from vanilla VAE and original data samples. Figure 9 shows some challenging examples for VAE to reconstruct. VAE-ACNF tends to reconstruct images by adding more details, not only to make it smoother, but also to possibly strengthen their identity of classes. Furthermore, due to the coarse variational approximation, some reconstructions of VAE fail to retain their features in original data and change the identity of classes.