Beyond Pathwise Derivatives: Policy Gradients from Trajectory Flows

Tolga Ok[1], Peyman Mohajerin Esfahani[1,2], and Mohamad Amin Sharifi Kolarijani[1]

[1]Delft University of Technology, The Netherlands
[2]University of Toronto, Canada

Abstract. Model-free policy gradient methods built on the score-function estimator have been widely successful in reinforcement learning. This success has not carried over to model-based estimators, which leverage a learned dynamics model to build more sample-efficient policy gradients. Such estimators typically differentiate through a learned model along the trajectory, and the resulting product of Jacobians quickly becomes unstable over long horizons. In this paper, we instead leverage a learned continuous flow, defined by an ordinary differential equation (ODE), that transports a simple base distribution onto the trajectory distribution. From the velocity of this flow alone, we form a transport-based policy gradient estimator that obviates solving the ODE and any long chain of Jacobians. Empirically, we show that this estimator is more accurate and stable than pathwise estimators built on learned dynamics models.

1. Introduction

Learning to transform a random variable with a known distribution into a random variable with a target distribution is useful in a surprisingly wide range of problems. Examples include density estimation, variational inference, and generative modeling [25]. In generative modeling, the target distribution is the data distribution known only through fixed samples. The goal is to learn a model of this distribution that can generate new samples and, ideally, evaluate their density. A common model transforms a random variable with a known distribution, such as a standard Gaussian, and sampling from the model then reduces to sampling from that distribution. This perspective underlies a broad range of generative models [25]. More recently, flow matching [20], rectified flow [21], and stochastic interpolants [1] have introduced a further family of closely related models of this kind. Here, we investigate using such a representation not only to generate samples, but also to estimate gradients of expectations when the target distribution depends on parameters.

In this work, we study a transport-based Monte Carlo gradient estimator for policy optimization that leverages a flow model with the aim of obtaining low-variance gradient estimates. More generally, consider an objective

$$J(\phi) = \mathbb{E}_{X_1 \sim p_\phi}\Big[f(X_1)\Big], \tag{1}$$

Date: November 16, 2026. Working draft, subject to substantial revision.
Correspondence to: Tolga Ok <t.ok@tudelft.nl>.

whose dependence on ϕ enters through the distribution p_ϕ. We seek a Monte Carlo estimate of $\partial_\phi J(\phi)$. Several estimator families can be used to estimate $\partial_\phi J(\phi)$, including *score-function*, *pathwise*, and *measure-valued* estimators, with different assumptions and variance properties [24]. Of particular interest here are pathwise estimators, which exploit a differentiable path from the parameters to the sampled outcome and apply the chain rule while holding the underlying randomness fixed. In model-based policy optimization, stochastic value gradient methods [14] and Dreamer [11] use such paths through differentiable dynamics models. Although pathwise estimators can offer lower variance than score-function estimators [24], applying the chain rule along long rollouts introduces products of transition Jacobians. These products can yield vanishing or exploding gradients, even with an accurate differentiable dynamics model [26, 34].

We instead investigate a transport-based representation of the policy-induced trajectory distribution, building on the connection between probability transport and pathwise gradient estimation [15, 27]. Here, a velocity field transforms a base noise sequence into a trajectory sample, and we study how its dependence on the policy parameters determines the gradient of expected return. The flow evolves in an auxiliary time variable that describes transport from noise to trajectories, rather than progression through environment transitions. A value function over this auxiliary time, of the kind that guides the fine-tuning of flow models [22], then supplies the derivative of the return, which we learn without solving the ODE. This allows us to formulate gradient estimation in terms of the transport dynamics, rather than differentiation through successive environment transitions.

Notation. For a set $\mathcal{X}$ in Euclidean space, $\Delta(\mathcal{X})$ denotes the set of probability distributions on $\mathcal{X}$, and $X \sim p$ denotes a random variable X with distribution $p \in \Delta(\mathcal{X})$, whose density we also denote by p. We write $\mathbb{E}_{X\sim p}[f(X)]$ for the expectation of $f(X)$, and $\mathcal{U}[0,1]$ for the uniform distribution on $[0,1]$. We write ∂_x for the partial derivative with respect to a variable x, which is the gradient for a scalar function and the Jacobian for a vector-valued function, and $\frac{d}{dx}$ for the total derivative with respect to x. For vectors a and b, $a \cdot b = a^\top b$ denotes the inner product and $\|a\|$ the Euclidean norm, and $\operatorname{div} F$ denotes the divergence of a vector field F.

2. Flow Matching

Let $\mathcal{X}_t \subset \mathcal{X}$ for all $t \in [0,1]$, let $\rho_0 \in \Delta(\mathcal{X}_0)$ and $p \in \Delta(\mathcal{X}_1)$, and let $X_0 \sim \rho_0$ and $X_1 \sim p$. We call p the *target distribution*, from which we can sample or of which we have a fixed set of samples, and ρ_0 the *base distribution*. We seek a smooth, invertible transformation that maps X_0 to a random variable whose distribution matches p. We are particularly interested in transformations where ρ_0 is a distribution with a tractable and easily computable density function, e.g., a standard Gaussian, from which we can sample. Our aim in this work is to use this transformation in the policy optimization problem to obtain Monte Carlo gradient estimates with respect to the policy parameters.

Although there is now a vast library of methods in the literature that learn a *flow* between ρ_0 and p, here we build on conditional flow matching (CFM) [20], which has shown great success recently in image synthesis [6] and robot control [3]. CFM builds a continuous normalizing flow model [5], that is, a velocity function whose corresponding ODE generates the flow, and it learns this velocity function

without ever solving the ODE throughout training. Assuming that there exists an underlying velocity function, we denote the trajectory it generates, e.g., through solving the ODE from $t = 0$ to $t = 1$, by $\psi_t : \mathcal{X}_0 \to \mathcal{X}_t$, and itself, i.e., the velocity at the current point, by $\dot{\psi}_t : \mathcal{X}_t \to \mathcal{X}$ for $t \in [0, 1]$, such that

$$\frac{d}{dt}\psi_t(x_0) = \dot{\psi}_t\big(\psi_t(x_0)\big), \qquad \psi_0(x_0) = x_0, \tag{2}$$

and $\psi_1(X_0)$ has the same distribution as X_1. Writing $X_t = \psi_t(X_0) \sim \rho_t$ with $\rho_t \in \Delta(\mathcal{X}_t)$, we say that $\dot{\psi}$ *generates* the *probability path* $(\rho_t)_{t\in[0,1]}$ [20]. The following proposition characterizes the probability path through the *continuity equation*.

Proposition 2.1 (Continuity equation)**.** *Let ρ_0 be continuously differentiable, and let $\dot{\psi}$ be continuous, twice continuously differentiable in x, and Lipschitz in x uniformly in t. Let ψ solve (2), let $X_0 \sim \rho_0$, and let ρ_t be probability densities on $\mathcal{X}$ for $t \in (0, 1]$, such that $\rho_t(x)$ is continuously differentiable in (t, x). Then $\psi_t(X_0) \sim \rho_t$ for every $t \in [0, 1]$ if and only if*

$$\partial_t \rho_t(x) + \operatorname{div}\big(\rho_t \dot{\psi}_t\big)(x) = 0 \quad \textit{for all} \quad t \in [0, 1],\ x \in \mathcal{X}. \tag{3}$$

For a proof, see Villani [32]. Importantly, the continuity equation in (3) appears not only in flow matching, where we obtain p via the learned flow, but also in Monte Carlo gradient estimation for policy optimization. To give a glimpse, the gradient of the objective function in (1) involves $\partial_\phi \rho_1$, and differentiating (3) in ϕ lets us substitute the time derivative $\partial_t \partial_\phi \rho_t$ with a divergence term. We then use the learned flow to obtain a gradient estimate, the details of which we explain later in the paper.

Conditional flow matching. Previously, we assumed that there exists an underlying flow ψ that maps X_0 to X_1. We opt to learn that map by a parameterized velocity function $\dot{\psi}^\theta$, e.g., through a regression objective given by

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_t\sim\rho_t}\left[\|\dot{\psi}^\theta_t(X_t) - \dot{\psi}_t(X_t)\|^2\right]. \tag{4}$$

However, we do not have access to the underlying flow $\dot{\psi}_t$ for $t \in [0, 1]$, otherwise it would defeat the purpose of learning one. Instead, CFM introduces a *conditional flow*, which we denote with a slight abuse of notation by $\psi_t(x_0; x_1)$, where x_1 is the condition and x_0 is the main variable. We denote by $p_{t|1}(\cdot|x_1)$ the *conditional probability path* that $\dot{\psi}_t(\cdot\,; x_1)$ generates from ρ_0. For a posterior distribution conditioned on x_t denoted by $p_{1|t}$, the underlying velocity is then given by

$$\dot{\psi}_t(x_t) = \mathbb{E}_{X_1\sim p_{1|t}}\left[\dot{\psi}_t(x_t; X_1)\right]. \tag{5}$$

The idea here is that even though we do not know underlying $\dot{\psi}_t(x)$, we can build $\dot{\psi}_t(x_t; x_1)$ for each x_1, and Equation (5) then gives a velocity that generates the mixture $\rho_t(x_t) = \mathbb{E}_{X_1\sim p}[p_{t|1}(x_t|X_1)]$ of the conditional paths [20, Thm. 1]. Leveraging Bayes' identity given by

$$\rho_t(x_t)p_{1|t}(x_1|x_t) = p(x_1)p_{t|1}(x_t|x_1), \tag{6}$$

CFM rewrites $\mathcal{L}_{\mathrm{FM}}(\theta)$ in (4) as an objective with the same gradient in θ that requires only $p_{t|1}$, which we can easily form, as opposed to ρ_t [20, Thm. 2]. For instance, given small $\sigma > 0$, $x_1 \sim p$, and

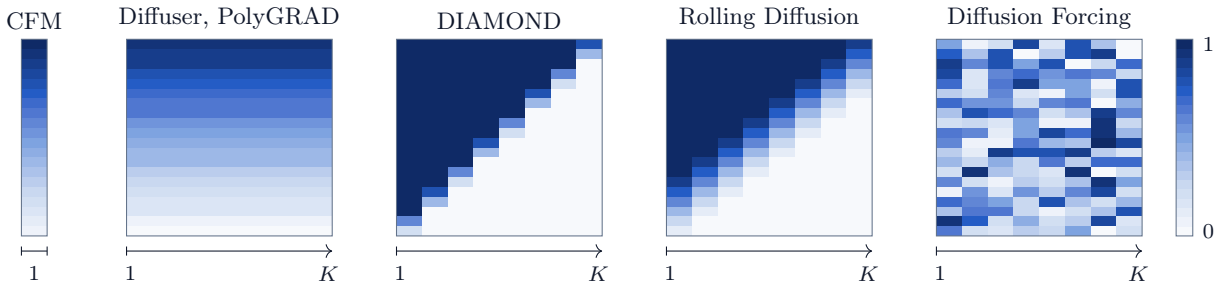


FIGURE 1. Each row is a vector of times $\tau = (\tau_1, \dots, \tau_K)$ that training samples at random, with the color showing each τ_k from 0 to 1. For Diffusion Forcing, the figure depicts rows whose times are sampled independently for each step.

$\rho_0 = \mathcal{N}(0, I)$, we may form $p_{t|1}(\cdot|x_1) = \mathcal{N}\big(tx_1, (1-(1-\sigma)t)^2 I\big)$. The CFM objective is then given by

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_1\sim p,\, X_t\sim p_{t|1}(\cdot|X_1)} \left[\|\dot\psi_t^\theta(X_t) - \dot\psi_t(X_t; X_1)\|^2 \right]. \tag{7}$$

There are several methods for defining $p_{t|1}$, including drawing a straight line between the two independent samples $x_0, x_1 \sim \rho_0 \otimes p$, in which case the trajectories of the underlying velocity $\dot\psi_t$ need not be straight, stochastic interpolants [1], which allow curved lines, and optimal transport conditional flow matching [31], which pairs a minibatch of x_0 and x_1 such that the close pairs are matched on average.

Random sequences. We now move our attention to the case where $X_1 = (X_1^1, \dots, X_1^K)$ is a Markov chain with K steps, and we split X_0 into K steps in the same way. Instead of a single time t, we assign each step k an individual time $\tau_k \in [0, 1]$ with $\tau = (\tau_1, \dots, \tau_K)$. We can model the generation of the chain by moving τ from $(0, \dots, 0)$ to $(1, \dots, 1)$ along a *schedule* $t \mapsto \tau(t)$, and categorize existing methods mainly by their schedules.

Figure 1 shows the schedules, where each row represents τ_k for the individual steps. Each method samples a row according to a distribution, e.g., uniform over rows. To give a contrast, we include CFM, which treats X_1 as a single step without sequential structure, i.e., $K = 1$. One extreme is moving all the indices together, i.e., $\tau = (t, \dots, t)$, as in Diffuser [16] and PolyGRAD [29]. This requires a single solution of the ODE, but each step moves while the past steps are still moving, and the velocity function must capture the dependence between steps from the points X_τ^j with $\tau_j < 1$. The opposite extreme moves one step at a time, as in autoregressive world models such as DIAMOND [2], and generates each step from a past whose times have reached one, as the chain itself does. The price is K consecutive solutions of the ODE, and the generation time thus grows with the length of the chain. Sliding windows, as in Rolling Diffusion [30], lie between the two, and Diffusion Forcing [4] samples the time of each step independently, such that a single model covers the rows of all of them.

3. Monte Carlo Gradient Estimation

For a distribution $p_\phi \in \Delta(\mathcal{X})$ parameterized by ϕ and a function $f\colon \mathcal{X} \to \mathbb{R}$, we seek to obtain an unbiased estimate of the gradient for the objective in (1). In policy optimization, the two

main approaches are the score-function estimator, also known as REINFORCE [33], and pathwise estimators, which are typically used in model-based reinforcement learning, including SVG [14] and Dreamer [11]. While the score-function estimator leverages the *log-trick*, and the pathwise estimator uses the *reparameterization trick* over a differentiable model, commonly learned from data, we turn to the transport-based estimator, which capitalizes on the *continuity equation* that we introduced in Section 2.

Similar to pathwise estimators, which require a differentiable model to form a path from the parameters ϕ to X_1, transport-based estimators rely on a velocity field $\dot{\psi}$ controlled by ϕ to obtain the gradient estimate. The following proposition defines the estimator.

Proposition 3.1 (Transport-based gradient)**.** *Let the conditional flow satisfy $\psi_1(x_0; x_1) = x_1$, and let $\dot{\psi}$ be the velocity in (5) for $p = p_\phi$, which generates $p_\phi = \rho_1$ from a base distribution ρ_0 independent of ϕ. If $\dot{\psi}$ is Lipschitz continuous in x, f is differentiable, and differentiation in ϕ and integration can be interchanged, then*

$$\partial_\phi J(\phi) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_0\sim\rho_0,\, X_1\sim p_\phi}\left[\partial_\phi \dot{\psi}_t(X_t)^\top\, \partial_x f_t(X_t)\right], \tag{8}$$

where $X_t = \psi_t(X_0; X_1)$ and f_t is the flow value of f given in Definition 3.2.

We emphasize here how the estimator in Proposition 3.1 differs from pathwise estimators. A pathwise estimator would use the flow as the differentiable model and differentiate through the solution $X_1 = \psi_1(X_0)$ of the ODE. The transport-based estimator instead differentiates the velocity only at a single time and point, and thus avoids solving the ODE, similar to CFM during training. The derivative $\partial_x f_t$, however, still involves the ODE, as the following definition makes precise.

Definition 3.2 (Flow value)**.** *Let a velocity $\dot{\psi}$ generate ρ_1 from a base distribution ρ_0. For $t \in [0, 1]$ and $x \in \mathcal{X}_t$, the* flow value *of f is*

$$f_t(x) = f(x_1), \tag{9}$$

where x_1 is the value at time 1 of the solution of (2) that passes through x at time t.

Evaluating $f_t(x)$ requires solving the ODE from x at time t to time 1, which determines the point at which f is evaluated. Differentiating through this solution, directly or by the adjoint method [5], gives $\partial_x f_t$ exactly. This would be akin to a pathwise gradient estimate, where the path between ϕ and X_1 is built through $\partial_x f_t$, which involves the solution of the ODE. However, in what follows, we learn f_t directly, which removes the need to solve the ODE in the transport-based estimator.

3.1. **Flow Value Gradient**

To learn f_t without solving the ODE, we rely on an identity stemming from the fact that the flow value is constant along each trajectory, i.e., $f_t(X_t) = f(X_1)$ for all $t \in [0, 1]$. However, we also note that the equality in f_t does not imply equality between $\partial_x f_t$ and $\partial_x f$, which is the term we aim to obtain. The following proposition characterizes f_t through the total derivative.

Proposition 3.3 (Transport equation). *Let $\dot{\psi}$ be continuously differentiable and Lipschitz continuous in x, and let f be continuously differentiable. Then the flow value f_t is the unique continuously differentiable solution of*

$$\partial_t f_t(x) + \partial_x f_t(x) \cdot \dot{\psi}_t(x) = 0, \qquad f_1 = f. \tag{10}$$

Unlike the definition of f_t, the left equality in (10) involves only the derivatives of f_t and the velocity at a single time and point, and can thus be obtained at sampled times and points without solving the ODE. This is akin to CFM, which also avoids the need for solving the ODE during training.

Intuitively, f_t carries the values of f backward along the trajectories, from time 1 to time t. The left identity in (10) is directly analogous to the Bellman equation with a terminal-only reward and the time of the flow in place of the steps of a sequence. Whereas temporal-difference learning fits a value function, e.g., a critic, using the Bellman equation over the MDP dynamics [19, 8, 10, 12], we fit f_t using the transport equation (10) over the flow dynamics. Similarly, VGG-Flow [22] fine-tunes a flow model with the gradient of a value function that satisfies (10) up to a control cost, and learns this gradient along trajectories obtained by solving the ODE.

Let f_t^ω be a model of the flow value with parameters ω approximating f_t. We construct f^ω such that $f_1^\omega = f$ for every ω, for instance, by

$$f_t^\omega(x) = f(x) + (1-t)\, g_t^\omega(x), \tag{11}$$

where g_t^ω is a learnable function, in which the terminal condition in (10) holds by construction. Consequently, we define the loss function following the left equation in (10) by

$$\mathcal{L}(\omega) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_0\sim\rho_0,\, X_1\sim p_\phi}\Big[\Big(g_t^\omega(X_t) - \mathrm{sg}\Big[\frac{d}{dt} f(X_t) + (1-t)\frac{d}{dt} g_t^\omega(X_t)\Big]\Big)^2\Big], \tag{12}$$

where $X_t = \psi_t(X_0; X_1)$ and sg denotes the *stop gradient* operator (see Appendix A.3 for the derivation). Replacing $\partial_x f_t$ by $\partial_x f_t^\omega$ in (8), we obtain a gradient estimate for ϕ, which is exact at $f_t^\omega = f_t$ and uses only samples of the distributions with which the estimator is defined, i.e., ρ_0 and p_ϕ. The learned velocity $\dot{\psi}^\theta$ enters the estimate only through $\partial_\phi \dot{\psi}_t$, which requires a model of how the velocity in (5) changes with ϕ, and Section 4 specifies this model for policy optimization.

3.2. **Relation to Standard Estimators**

The two widely used estimators [24], namely, the score-function and pathwise estimators, express $\partial_\phi J(\phi)$ through different representations of $\partial_\phi p_\phi$. To be precise, the score-function estimator [33] uses the log density identity $\partial_\phi p_\phi = p_\phi(\partial_\phi \log p_\phi)$, whereas the pathwise estimator [18] uses a differentiable map g_ϕ that generates $X_1 = g_\phi(X_0)$ from $X_0 \sim \rho_0$ independent of ϕ. On the other hand, the transport-based estimator in (8) leverages a differentiable velocity function $\dot{\psi}$ to obtain the estimate.

We summarize the three estimators in the following.

$$\partial_\phi J(\phi) = \mathbb{E}_{X\sim p_\phi}\big[f(X)\,\partial_\phi \log p_\phi(X)\big] \qquad \text{(score function)} \tag{13a}$$

$$= \mathbb{E}_{Z\sim\rho}\big[\partial_\phi g_\phi(Z)^\top\,\partial_x f\big(g_\phi(Z)\big)\big] \qquad \text{(pathwise)} \tag{13b}$$

$$= \mathbb{E}_{t\sim\mathcal{U}[0,1],\,X_0\sim\rho_0,\,X_1\sim p_\phi}\big[\partial_\phi \dot\psi_t(X_t)^\top\,\partial_x f_t(X_t)\big] \qquad \text{(transport-based)} \tag{13c}$$

where $X_t = \psi_t(X_0; X_1)$ and ρ is independent of ϕ. We note that all three estimators are unbiased and, hence, equivalent in expectation. However, this holds for the pathwise and transport-based estimators only when g_ϕ and $\dot\psi_t$ for all $t \in [0,1]$, respectively, are exact, and the transport-based estimator also requires the exact flow value f_t. The score-function estimator differentiates neither f nor the generation of X, which allows both to be black boxes, but, as a rule of thumb, it has the highest variance [24, Sec. 3]. The map g_ϕ can be the solution at time 1 of an ODE with a velocity that depends on ϕ, as in neural ODEs [5], or a function that maps Z to X directly, as in Kingma and Welling [18]. In both cases, the pathwise estimator requires a differentiable path from ϕ to X, e.g., by differentiating through the ODE solution in the former case and through the map in the latter case. More commonly, in sequential settings, i.e., over Markov chains, pathwise estimators are built over an approximate model of the dynamics [14, 11], which results in differentiating through the steps of the dynamics, and $\partial_\phi g_\phi$ then multiplies the Jacobians of all steps. This product quickly becomes unstable as the number of steps grows [26, 34, 23]. Moreover, this model has to predict every transition, and rolling it out compounds its errors over the steps [17]. A generative model of whole trajectories, trained for the accuracy of the generated trajectories rather than of single steps, avoids these compounding errors and scales better with the number of steps [16].

The pathwise and transport-based estimators both weight the derivative in ϕ by a derivative of f, and both require a model of how X_1 is generated, the map g_ϕ or the velocity field $\dot\psi$. In that sense, the transport-based estimator generalizes the pathwise estimator, where the solution of the ODE is the path, and $\partial_x f_t$ along this path is the adjoint state of the ODE [5], which the pathwise estimator computes by solving the ODE backward. However, here we aim to avoid solving the ODE in training, which can be unstable and costly [9, 35, 7]. The transport-based estimator requires only the distribution of X_t, and obtains $\partial_x f_t$ from the flow value, which Section 3.1 learns from samples of the conditional flow alone, instead of propagating $\partial_x f$ backward along the ODE.

4. Policy Optimization

In this section, we introduce a policy optimization scheme built on the transport-based estimator and flow matching. So far, we have not addressed how the policy parameters ϕ influence the velocity field, nor have we specified X (or X_t) in a particular problem, i.e., in policy optimization. Here, we make the relation concrete by conditioning the velocity field on a parametric policy. To this end, we define Markov Decision Processes (MDPs) in the following.

Markov Decision Processes. We model the policy optimization problem on an MDP [28], which we represent by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathbb{P}, r, \mu, \gamma\rangle$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the action space, $\mathbb{P}\colon \mathcal{S}\times\mathcal{A} \to \Delta(\mathcal{S})$ denotes the transition kernel, and $\gamma \in [0,1)$ denotes the discount factor.

Starting from an initial state s_0 with an initial distribution $\mu \in \Delta(\mathcal{S})$, at each step $k \in \mathbb{N}$, an action $a_k \sim \pi_\phi(\cdot \mid s_k)$ is drawn from a policy $\pi_\phi \colon \mathcal{S} \to \Delta(\mathcal{A})$ with parameters $\phi \in \Phi$, and the next state is drawn as $s_{k+1} \sim \mathbb{P}(\cdot \mid s_k, a_k)$. The resulting trajectory, which we denote by $X_1 = (S_0, A_0, S_1, \dots)$, is distributed according to p_ϕ. Given a pair (s_k, a_k), we define the intermediate reward by $r(s_k, a_k)$.

The policy optimization objective has the form of Equation (1) with $f(X_1) = \sum_{k \geq 0} \gamma^k r(S_k, A_k)$. To maximize it, we follow a stochastic gradient ascent scheme, which requires a gradient estimate with respect to ϕ of the objective given by

$$\partial_\phi J(\phi) = \partial_\phi \mathbb{E}_{X_1 \sim p_\phi} \left[\sum_{k \geq 0} \gamma^k r(S_k, A_k) \right]. \tag{14}$$

We note that the expectation in Equation (14) is reminiscent of the standard formulation in the reinforcement learning literature, where the expectation is taken of a value function over the initial state. We omit the details of such connections, but emphasize that in score-function estimators, value functions play the role of a variance reduction technique known as a control variate [24, Sec. 7.4]. Similar variance reduction schemes, such as conditioning, i.e., Rao-Blackwellization [24, Sec. 7.3], and coupling [24, Sec. 7.2], may apply to all three estimators covered in the previous section. For instance, in the absence of a differentiable reward function r, Rao-Blackwellization provides an alternative objective if instead we have access to the probability density function over the reward function. For detailed coverage, we refer the reader to Mohamed et al. [24].

References

[1] Albergo, M. S. and Vanden-Eijnden, E. (2023). Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*.

[2] Alonso, E., Jelley, A., Micheli, V., Kanervisto, A., Storkey, A., Pearce, T., and Fleuret, F. (2024). Diffusion for world modeling: Visual details matter in Atari. In *Advances in Neural Information Processing Systems*.

[3] Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Shi, L. X., Tanner, J., Vuong, Q., Walling, A., Wang, H., and Zhilinsky, U. (2025). π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems*.

[4] Chen, B., Martí Monsó, D., Du, Y., Simchowitz, M., Tedrake, R., and Sitzmann, V. (2024). Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems*.

[5] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31.

[6] Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Lacey, K., Goodwin, A., Marek, Y., and Rombach, R. (2024). Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 12606–12633.

[7] Finlay, C., Jacobsen, J.-H., Nurbekyan, L., and Oberman, A. M. (2020). How to train your neural ODE: The world of Jacobian and kinetic regularization. In *International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3154–3164.

[8] Fujimoto, S., van Hoof, H., and Meger, D. (2018). Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596.

[9] Gholami, A., Keutzer, K., and Biros, G. (2019). ANODE: Unconditionally accurate memory-efficient gradients for neural ODEs. In *International Joint Conference on Artificial Intelligence*, pages 730–736.

[10] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870.

[11] Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2020). Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*.

[12] Hansen, N., Su, H., and Wang, X. (2024). TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*.

[13] Hartman, P. (2002). *Ordinary Differential Equations*, volume 38 of *Classics in Applied Mathematics*. SIAM, 2 edition.

[14] Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. (2015). Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, volume 28.

[15] Jankowiak, M. and Obermeyer, F. (2018). Pathwise derivatives beyond the reparameterization trick. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2235–2244.

[16] Janner, M., Du, Y., Tenenbaum, J. B., and Levine, S. (2022). Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9902–9915.

[17] Janner, M., Fu, J., Zhang, M., and Levine, S. (2019). When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 32.

[18] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *International Conference on Learning Representations*.

[19] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2016). Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*.

[20] Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. (2023). Flow matching for generative modeling. In *International Conference on Learning Representations*.

[21] Liu, X., Gong, C., and Liu, Q. (2023). Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*.

[22] Liu, Z., Xiao, T. Z., Domingo-Enrich, C., Liu, W., and Zhang, D. (2025). Value gradient guidance for flow matching alignment. In *Advances in Neural Information Processing Systems*, volume 38.

[23] Ma, M., Ni, T., Gehring, C., D'Oro, P., and Bacon, P.-L. (2024). Do transformer world models give better policy gradients? In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 33855–33879.

[24] Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. (2020). Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62.

[25] Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64.

[26] Parmas, P., Rasmussen, C. E., Peters, J., and Doya, K. (2018). PIPPS: Flexible model-based policy search robust to the curse of chaos. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4065–4074.

[27] Parmas, P. and Sugiyama, M. (2021). A unified view of likelihood ratio and reparameterization gradients. In *International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 4078–4086.

[28] Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

[29] Rigter, M., Yamada, J., and Posner, I. (2024). World models via policy-guided trajectory diffusion. *Transactions on Machine Learning Research*.

[30] Ruhe, D., Heek, J., Salimans, T., and Hoogeboom, E. (2024). Rolling diffusion models. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 42818–42835.

[31] Tong, A., Fatras, K., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Wolf, G., and Bengio, Y. (2024). Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*.

[32] Villani, C. (2009). *Optimal Transport: Old and New*, volume 338 of *Grundlehren der mathematischen Wissenschaften*. Springer.

[33] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256.

[34] Xu, J., Makoviychuk, V., Narang, Y., Ramos, F., Matusik, W., Garg, A., and Macklin, M. (2022). Accelerated policy learning with parallel differentiable simulation. In *International Conference on Learning Representations*.

[35] Zhuang, J., Dvornek, N., Li, X., Tatikonda, S., Papademetris, X., and Duncan, J. (2020). Adaptive checkpoint adjoint method for gradient estimation in neural ODE. In *International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11639–11649.

Appendix A. Derivations

A.1. The Conditional Flow Matching Objective

We derive the conditional flow matching objective (7) from the marginal objective (4), using the notation of Section 2. We first decompose the squared error at fixed time and position, then average over the joint distribution to show that the two losses differ by a constant independent of θ.

Throughout, we consider the joint law

$$t \sim \mathcal{U}[0,1], \qquad X_1 \sim p, \qquad X_t \sim p_{t|1}(\cdot|X_1).$$

We assume that the velocities have finite second moments under this law, and that the target distribution and conditional paths do not depend on θ.

Conditional squared-error decomposition. For fixed t and x_t, Equation (5) expresses the marginal velocity as the posterior mean

$$\dot{\psi}_t(x_t) = \mathbb{E}_{X_1 \sim p_{1|t}(\cdot|x_t)}\left[\dot{\psi}_t(x_t; X_1)\right], \qquad p_{1|t}(x_1|x_t) = \frac{p(x_1)p_{t|1}(x_t|x_1)}{\rho_t(x_t)},$$

where the posterior is defined whenever $\rho_t(x_t) > 0$. All expectations in the following calculation are over this posterior. Adding and subtracting $\dot{\psi}_t(x_t)$ inside the squared norm gives

$$\begin{aligned}\mathbb{E}\left[\|\dot{\psi}_t^\theta(x_t) - \dot{\psi}_t(x_t; X_1)\|^2\right] &= \|\dot{\psi}_t^\theta(x_t) - \dot{\psi}_t(x_t)\|^2 + \mathbb{E}\left[\|\dot{\psi}_t(x_t) - \dot{\psi}_t(x_t; X_1)\|^2\right] \\ &\quad + 2\big(\dot{\psi}_t^\theta(x_t) - \dot{\psi}_t(x_t)\big) \cdot \mathbb{E}\left[\dot{\psi}_t(x_t) - \dot{\psi}_t(x_t; X_1)\right].\end{aligned}$$

The cross term vanishes, since Equation (5) gives

$$\mathbb{E}\left[\dot{\psi}_t(x_t) - \dot{\psi}_t(x_t; X_1)\right] = 0.$$

We therefore obtain the conditional squared-error decomposition

$$\mathbb{E}\left[\|\dot{\psi}_t^\theta(x_t) - \dot{\psi}_t(x_t; X_1)\|^2\right] = \|\dot{\psi}_t^\theta(x_t) - \dot{\psi}_t(x_t)\|^2 + \mathbb{E}\left[\|\dot{\psi}_t(x_t; X_1) - \dot{\psi}_t(x_t)\|^2\right]. \tag{15}$$

Averaging over the joint distribution. We now average (15) over $t \sim \mathcal{U}[0,1]$ and $X_t \sim \rho_t$. By Bayes' identity (6), the joint law obtained by drawing X_t first and then X_1 from the posterior equals the joint law obtained by drawing $X_1 \sim p$ first and then $X_t \sim p_{t|1}(\cdot|X_1)$. Consequently,

$$\begin{aligned}&\mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_t\sim\rho_t}\mathbb{E}_{X_1\sim p_{1|t}(\cdot|X_t)}\left[\|\dot{\psi}_t^\theta(X_t) - \dot{\psi}_t(X_t; X_1)\|^2\right] \\ &\quad = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_1\sim p,\, X_t\sim p_{t|1}(\cdot|X_1)}\left[\|\dot{\psi}_t^\theta(X_t) - \dot{\psi}_t(X_t; X_1)\|^2\right] = \mathcal{L}_{\mathrm{CFM}}(\theta).\end{aligned}$$

The first term on the right of (15) averages to $\mathcal{L}_{\mathrm{FM}}(\theta)$, while the second gives a nonnegative constant. Hence,

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathcal{L}_{\mathrm{FM}}(\theta) + C, \qquad C = \mathbb{E}\left[\|\dot{\psi}_t(X_t; X_1) - \dot{\psi}_t(X_t)\|^2\right] \geq 0, \tag{16}$$

where the expectation defining C is under the joint law specified above.

Consequences for training. Since C is independent of θ, the losses have the same minimizers and, whenever differentiation under the expectation is valid, the same gradient,

$$\partial_\theta \mathcal{L}_{\mathrm{CFM}}(\theta) = \partial_\theta \mathcal{L}_{\mathrm{FM}}(\theta),$$

as established by Lipman et al. [20, Thm. 2]. Unlike the marginal objective, CFM requires neither evaluating the marginal velocity nor sampling from the posterior $p_{1|t}$, and requires only target samples and a tractable conditional path and velocity.

A.2. **The Transport-Based Gradient**

This subsection proves Proposition 3.1. We state the conditions, express the probability path through samples of the conditional flow, and prove Proposition 3.1 from two lemmas, which we prove at the end.

Conditions. For a vector ϕ, the derivative $\partial_\phi \rho_t$ is a vector, $\partial_\phi \dot\psi_t$ is the Jacobian of $\dot\psi_t$ in ϕ, and div acts on each column of a matrix, such that the equations below hold coordinate by coordinate in ϕ. For every ϕ in an open set, ρ_0 and $\dot\psi$ satisfy the assumptions of Proposition 2.1, ρ_0 is positive and does not depend on ϕ, and ρ_t is the density of $X_t = \psi_t(X_0)$ for $X_0 \sim \rho_0$. The velocity $\dot\psi$ is the velocity in (5) for $p = p_\phi$, and the conditional flow satisfies $\psi_0(x_0; x_1) = x_0$ and $\psi_1(x_0; x_1) = x_1$. The mixture in (17) is continuously differentiable in (t, x), and the expectation of the mixture over X_1 can be exchanged with the derivatives in t and x. The functions $\dot\psi_t(x)$ and $\rho_t(x)$ are twice continuously differentiable in (t, x, ϕ), f is continuously differentiable, and $f_t(x)$ is continuously differentiable in (t, x). The integrals below are finite, can be differentiated under the integral sign, and can be integrated by parts without boundary terms, which holds when $f_t \rho_t \dot\psi_t$, $f_t\, \partial_\phi \rho_t\, \dot\psi_t$ and $f_t \rho_t\, \partial_\phi \dot\psi_t$ decay fast enough at infinity.

Samples of the probability path. Let $X_0 \sim \rho_0$ and $X_1 \sim p_\phi$ be independent, and let $X_t = \psi_t(X_0; X_1)$. Given $X_1 = x_1$, the point X_t has the density $p_{t|1}(\cdot|x_1)$, and X_t thus has the density

$$\tilde\rho_t(x) = \mathbb{E}_{X_1 \sim p_\phi}\big[p_{t|1}(x|X_1)\big]. \tag{17}$$

For $t \in [0, 1)$, Proposition 2.1 gives the continuity equation $\partial_t p_{t|1}(\cdot|x_1) + \operatorname{div}\big(p_{t|1}(\cdot|x_1)\, \dot\psi_t(\cdot\,; x_1)\big) = 0$ for each conditional path, and taking the expectation over $X_1 \sim p_\phi$ gives

$$\partial_t \tilde\rho_t(x) + \operatorname{div}\Big(\mathbb{E}_{X_1 \sim p_\phi}\big[p_{t|1}(x|X_1)\, \dot\psi_t(x; X_1)\big]\Big) = 0.$$

Bayes' identity (6) with the mixture $\tilde\rho_t$ turns the expectation into $\tilde\rho_t(x)\, \mathbb{E}_{X_1 \sim p_{1|t}(\cdot|x)}[\dot\psi_t(x; X_1)]$, which equals $\tilde\rho_t(x)\, \dot\psi_t(x)$ by (5). The mixture $\tilde\rho_t$ thus satisfies the continuity equation (3) for $t \in [0, 1)$, and by continuity also at $t = 1$, and $\psi_0(x_0; x_1) = x_0$ gives $\tilde\rho_0 = \rho_0$. Proposition 2.1 then gives $\psi_t(X_0) \sim \tilde\rho_t$, that is, $\tilde\rho_t = \rho_t$ for every $t \in [0, 1]$. At $t = 1$, the condition $\psi_1(x_0; x_1) = x_1$ gives $\psi_1(X_0; X_1) = X_1 \sim p_\phi$, and thus $\rho_1 = p_\phi$. For a function h on $\mathcal{X}$, the point $X_t = \psi_t(X_0; X_1)$ therefore satisfies

$$\mathbb{E}_{X_0 \sim \rho_0,\, X_1 \sim p_\phi}\big[h(X_t)\big] = \mathbb{E}_{X_t \sim \rho_t}\big[h(X_t)\big]. \tag{18}$$

The two points $\psi_t(X_0; X_1)$ and $\psi_t(X_0)$ differ in general, and only their distributions agree.

Proof of Proposition 3.1. Since $\rho_1 = p_\phi$, the gradient of the objective is

$$\partial_\phi J(\phi) = \int_{\mathcal{X}} f(x)\, \partial_\phi \rho_1(x)\, dx. \tag{19}$$

The derivative $\partial_\phi \rho_t$ evolves by the continuity equation differentiated in ϕ (Lemma A.1),

$$\partial_t \partial_\phi \rho_t + \operatorname{div}\big(\partial_\phi \rho_t \, \dot{\psi}_t\big) = -\operatorname{div}\big(\rho_t \, \partial_\phi \dot{\psi}_t\big), \tag{20}$$

and the flow value evolves by the transport equation (Proposition 3.3),

$$\partial_t f_t + \partial_x f_t \cdot \dot{\psi}_t = 0, \qquad f_1 = f. \tag{21}$$

The function

$$I(t) = \int_{\mathcal{X}} f_t(x) \, \partial_\phi \rho_t(x) \, dx \tag{22}$$

connects the two ends of the flow. At $t = 1$, the condition $f_1 = f$ and Equation (19) give $I(1) = \partial_\phi J(\phi)$, and at $t = 0$, the base distribution ρ_0 does not depend on ϕ, which gives $I(0) = 0$. Differentiating I with (20) and (21) leaves only the source term of (20) (Lemma A.2),

$$I'(t) = \mathbb{E}_{X_t \sim \rho_t}\big[\partial_\phi \dot{\psi}_t(X_t)^\top \, \partial_x f_t(X_t)\big]. \tag{23}$$

Integrating over t gives

$$\partial_\phi J(\phi) = I(1) - I(0) = \int_0^1 I'(t) \, dt = \mathbb{E}_{t \sim \mathcal{U}[0,1],\, X_t \sim \rho_t}\big[\partial_\phi \dot{\psi}_t(X_t)^\top \, \partial_x f_t(X_t)\big].$$

The integrand is a function of X_t alone, and (18) gives (8). The flow value is what removes $\partial_\phi \rho_t$ from (23). With f in place of f_t in (22), the derivative $I'(t)$ would retain the term $\int_{\mathcal{X}} \partial_\phi \rho_t \, \partial_x f \cdot \dot{\psi}_t \, dx$, which the transport equation in (21) cancels.

Lemma A.1 (Derivative of the probability path)**.** *The derivative $\partial_\phi \rho_t$ satisfies* (20)*.*

Proof. The continuity equation (3) holds for every ϕ in an open set, and the terms of (3) are continuously differentiable in ϕ. Differentiating it in ϕ, exchanging the order of the derivatives in t and ϕ, and applying the product rule to $\partial_\phi(\rho_t \dot{\psi}_t)$ gives (20). □

Lemma A.2 (Source term)**.** *The function I in* (22) *satisfies* (23)*.*

Proof. By the product rule, (21) and (20),

$$I'(t) = -\int_{\mathcal{X}} \big(\partial_x f_t \cdot \dot{\psi}_t\big) \, \partial_\phi \rho_t \, dx - \int_{\mathcal{X}} f_t \, \operatorname{div}\big(\partial_\phi \rho_t \, \dot{\psi}_t + \rho_t \, \partial_\phi \dot{\psi}_t\big) \, dx.$$

Integrating the second integral by parts, $\int f \operatorname{div} F \, dx = -\int F^\top \partial_x f \, dx$, turns it into $\int_{\mathcal{X}} \big(\partial_\phi \rho_t \, \partial_x f_t \cdot \dot{\psi}_t + \rho_t \, \partial_\phi \dot{\psi}_t^\top \partial_x f_t\big) \, dx$. The first term of this integral cancels the first integral of $I'(t)$, and the second term is the right side of (23). □

A.3. **The Flow Value**

This subsection proves Proposition 3.3 and shows that the gradient of the loss (12) in ω is the gradient with the velocity in place of the conditional velocity. We state the conditions, prove the proposition from one lemma, and then turn to the loss.

Conditions. The velocity $\dot{\psi}_t(x)$ is continuous in (t, x), continuously differentiable in x, and Lipschitz continuous in x uniformly in t, and f is continuously differentiable. For $t \in [0, 1]$ and $x \in \mathcal{X}$, the trajectory x_s for $s \in [0, 1]$ with $x_t = x$ is then unique, and x_1 is a continuously differentiable function

of (t, x) [13]. For the loss, $g_t^\omega(x)$ is continuously differentiable in (t, x, ω), the conditional flow satisfies $\frac{d}{dt}\psi_t(x_0; x_1) = \dot{\psi}_t(\psi_t(x_0; x_1); x_1)$, and $\dot{\psi}$ is the velocity in (5) for $p = p_\phi$. The expectations below are finite and can be differentiated in ω under the integral sign.

Lemma A.3 (Flow value along the flow)**.** *Let x_s for $s \in [0, 1]$ solve (2). Then $f_s(x_s) = f(x_1)$ for every $s \in [0, 1]$, and $f_t(x)$ is continuously differentiable in (t, x).*

Proof. The solution of (2) through x_s at time s is unique and is thus $x_\cdot$ itself, and (9) gives $f_s(x_s) = f(x_1)$. The flow value is the composition of f with the continuously differentiable function $(t, x) \mapsto x_1$, and is thus continuously differentiable. □

Proof of Proposition 3.3. Along a trajectory x_s, Lemma A.3 gives $f_s(x_s) = f(x_1)$ for every $s \in [0, 1]$, which is constant in s. The chain rule at $s = t$ gives $\partial_t f_t(x_t) + \partial_x f_t(x_t) \cdot \dot{\psi}_t(x_t) = 0$, and $s = 1$ gives $f_1 = f$. Every point of $\mathcal{X}_t$ lies on a trajectory at time t, and f_t thus solves (10). If h_t is another continuously differentiable solution, the chain rule and Equation (10) give $\frac{d}{ds}h_s(x_s) = 0$, and thus $h_t(x_t) = h_1(x_1) = f(x_1) = f_t(x_t)$, which gives $h_t = f_t$.

The loss. The bracket in (12) exceeds $g_t^\omega(X_t)$ by $\frac{d}{dt}f(X_t) - g_t^\omega(X_t) + (1 - t)\frac{d}{dt}g_t^\omega(X_t) = \frac{d}{dt}f_t^\omega(X_t)$, and the stopped gradient gives

$$\partial_\omega \mathcal{L}(\omega) = -2\,\mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_0\sim\rho_0,\, X_1\sim p_\phi}\left[\frac{d}{dt}f_t^\omega(X_t)\,\partial_\omega g_t^\omega(X_t)\right]. \tag{24}$$

The chain rule and the conditions on the conditional flow give $\frac{d}{dt}f_t^\omega(X_t) = \partial_t f_t^\omega(X_t) + \partial_x f_t^\omega(X_t) \cdot \dot{\psi}_t(X_t; X_1)$. The point $X_t = \psi_t(X_0; X_1)$ has the distribution $p_{t|1}(\cdot|X_1)$ given X_1, and Bayes' identity in (6) thus gives $X_t \sim \rho_t$ and $X_1 \sim p_{1|t}(\cdot|X_t)$ given X_t. In (24), only $\dot{\psi}_t(X_t; X_1)$ depends on X_1 given X_t, and it enters linearly. Taking the conditional expectation given X_t and applying (5) gives

$$\partial_\omega \mathcal{L}(\omega) = -2\,\mathbb{E}_{t\sim\mathcal{U}[0,1],\, X_t\sim\rho_t}\left[\left(\partial_t f_t^\omega(X_t) + \partial_x f_t^\omega(X_t) \cdot \dot{\psi}_t(X_t)\right)\partial_\omega g_t^\omega(X_t)\right].$$

At $f^\omega = f_t$, the bracket vanishes by Proposition 3.3, and $\partial_\omega \mathcal{L}(\omega) = 0$. Conversely, if the bracket vanishes for every t and x, then f^ω solves (10), and Proposition 3.3 gives $f^\omega = f_t$. Without the stopped gradient, the loss would be the expectation of $(\frac{d}{dt}f_t^\omega(X_t))^2$ under the same distribution, and the conditional variance of $\partial_x f_t^\omega(X_t) \cdot \dot{\psi}_t(X_t; X_1)$ given X_t would add a term that depends on ω. Forward-mode automatic differentiation computes $\frac{d}{dt}f_t^\omega(X_t)$, the derivative of f^ω in the direction $(1, \dot{\psi}_t(X_t; X_1))$ of time and space, as one Jacobian-vector product.