Path gradients for Normalizing Flows

Lorenz Vaitl

Lattice Field Theory

- Target

$$p(x) = \frac{1}{Z} e^{-S(x)}$$

Lattice Field Theory

- Target

$$p(x) = \frac{1}{Z} e^{-S(x)}$$

  - Intractable
  - Known in closed form

Lattice Field Theory

- Target

$$p(x) = \frac{1}{Z} e^{-S(x)}$$

  - Intractable
  - Known in closed form

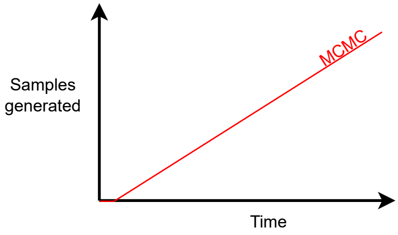
- Highly symmetric problems

Lattice Field Theory

- Target

$$p(x) = \frac{1}{Z} e^{-S(x)}$$

  - Intractable
  - Known in closed form
- Highly symmetric problems
- Currently $p(x)$ is sampled using MCMC

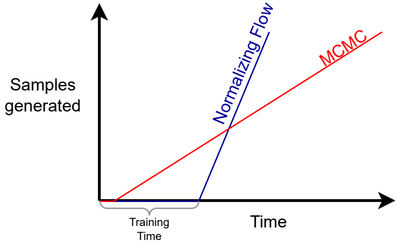


Lattice Field Theory

- Target

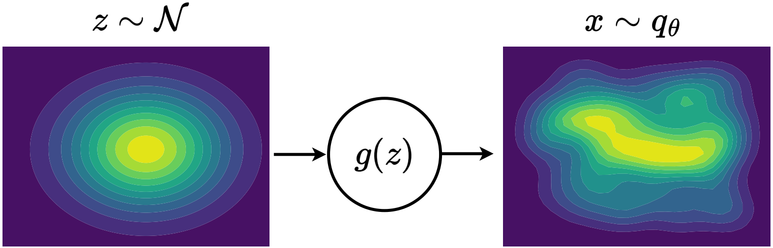
$$p(x) = \frac{1}{Z} e^{-S(x)}$$

  - Intractable
  - Known in closed form
- Highly symmetric problems
- Currently $p(x)$ is sampled using MCMC
- Deep Generative Models could help speed up sampling



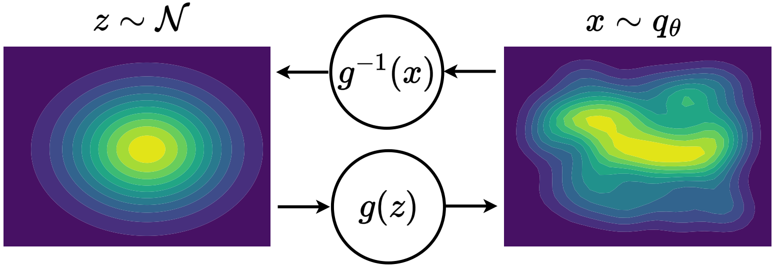
Normalizing Flows

$$\log q_\theta(g_\theta(z)) = \log q_0\left(z\right) - \log\left|\det \frac{\partial g_\theta(z)}{\partial z}\right|$$



Normalizing Flows

$$
\begin{aligned}
\log q_\theta(g_\theta(z)) &= \log q_0\left(z\right) - \log\left|\det \frac{\partial g_\theta(z)}{\partial z}\right| \\
&= \left.\left(\log q_0\left(g_\theta^{-1}(x)\right) + \log\left|\det \frac{\partial g_\theta^{-1}(x)}{\partial x}\right|\right)\right|_{x=g_\theta(z)}
\end{aligned}
$$



How to Normalizing Flow & Simulation Based Inference

- Correct for bias in sampling with (self-normalized) importance sampling

$$\mathbb{E}_{x\sim p}\left[\mathcal{Q}(x)\right] = \mathbb{E}_{x\sim q_\theta}\Big[\underbrace{\frac{p(x)}{q_\theta(x)}}_{w(x)}\,\mathcal{Q}(x)\Big]$$

How to Normalizing Flow & Simulation Based Inference

- Correct for bias in sampling with (self-normalized) importance sampling

$$\mathbb{E}_{x\sim p}\left[\mathcal{Q}(x)\right] = \mathbb{E}_{x\sim q_\theta}\Big[\underbrace{\frac{p(x)}{q_\theta(x)}}_{w(x)}\,\mathcal{Q}(x)\Big]$$

- Minimize KL divergence

$$KL(q_\theta|p) = \mathbb{E}_{x\sim q_\theta}\left[\log q_\theta(x) - \log p(x)\right]$$

How to Normalizing Flow & Simulation Based Inference

- Correct for bias in sampling with (self-normalized) importance sampling

$$\mathbb{E}_{x\sim p}\left[\mathcal{Q}(x)\right] = \mathbb{E}_{x\sim q_\theta}\Big[\underbrace{\frac{p(x)}{q_\theta(x)}}_{w(x)}\mathcal{Q}(x)\Big]$$

- Minimize KL divergence

$$KL(q_\theta|p) = \mathbb{E}_{x\sim q_\theta}\left[\log q_\theta(x) - \log p(x)\right]$$

- KL divergence is typically minimized using the reparametrization trick

$$\frac{\mathrm{d}KL}{\mathrm{d}\theta} = \mathbb{E}_{z\sim q_0}\left[\frac{\mathrm{d}}{\mathrm{d}\theta}\left(\log q_\theta(g_\theta(z)) - \log p(g_\theta(z))\right)\right]$$

Path gradient estimators

Path gradient estimators

- Proposed by [Roeder et al., 2017]

Path gradient estimators

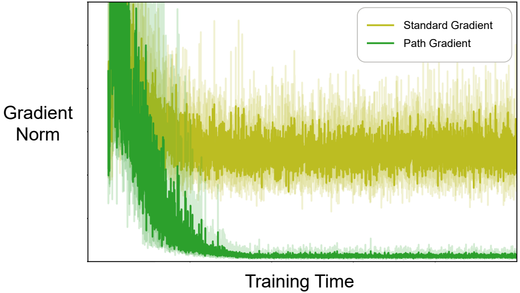
- Proposed by [Roeder et al., 2017]
- Sticking the landing

Path gradient estimators

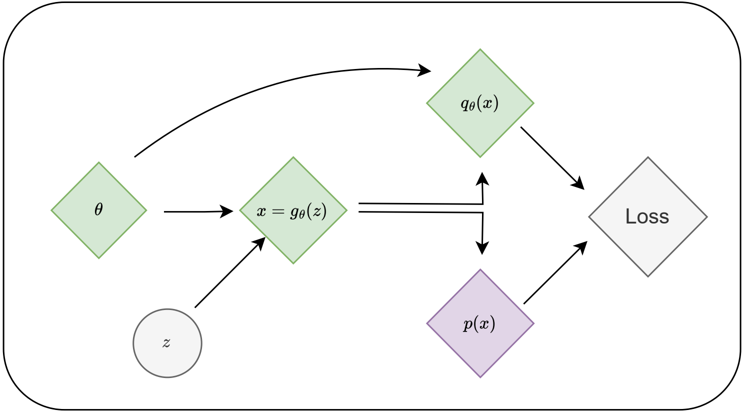
- Proposed by [Roeder et al., 2017]
- Sticking the landing
- Low Variance especially for perfect approximation

Path gradient estimators

- Proposed by [Roeder et al., 2017]
- Sticking the landing
- Low Variance especially for perfect approximation

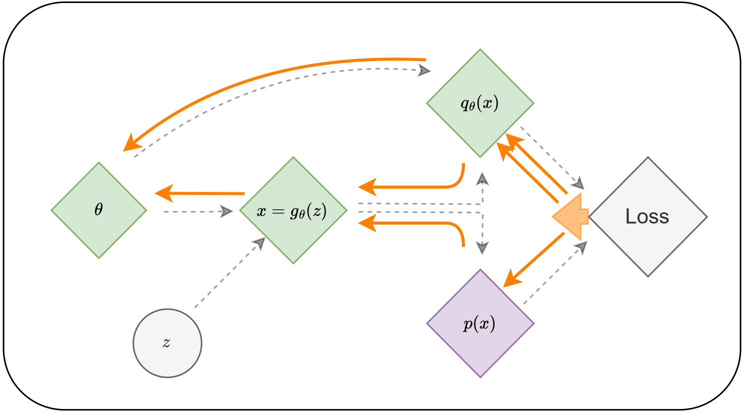


What are path gradients?



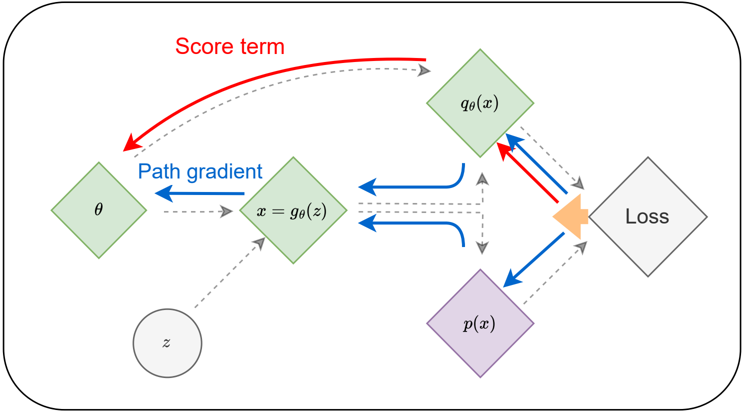
What are path gradients?

$$\frac{\mathrm{d}}{\mathrm{d}\theta}\left(\log q_\theta(g_\theta(z)) - \log p(g_\theta(z))\right)$$



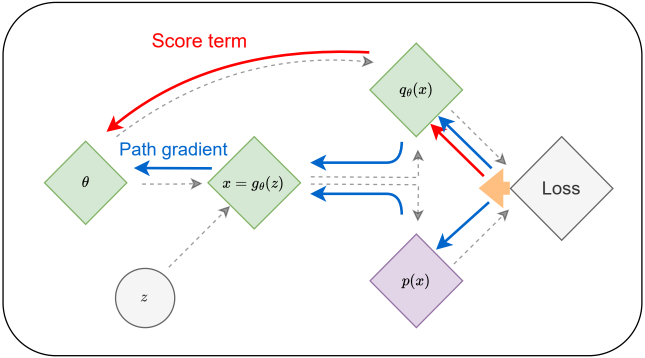
What are path gradients?

$$\frac{\mathrm{d}}{\mathrm{d}\theta}\left(\log q_\theta(g_\theta(z)) - \log p(g_\theta(z))\right) = \frac{\partial \log q_\theta(g_\theta(z)) - \log p(g_\theta(z))}{\partial g_\theta(z)} \frac{\partial g_\theta(z)}{\partial \theta} + \frac{\partial \log q_\theta(x)}{\partial \theta}\bigg|_{x=g_\theta(z)}$$



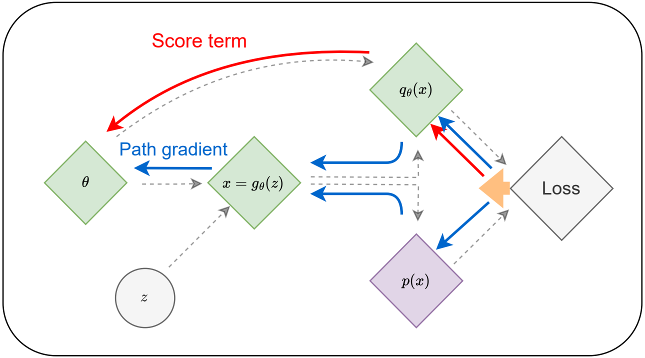
Total Gradient = **Path Gradient** + **Score Term**

Most widely used *Real Information* *Noise*



Total Gradient = **Path Gradient** + **Score Term**

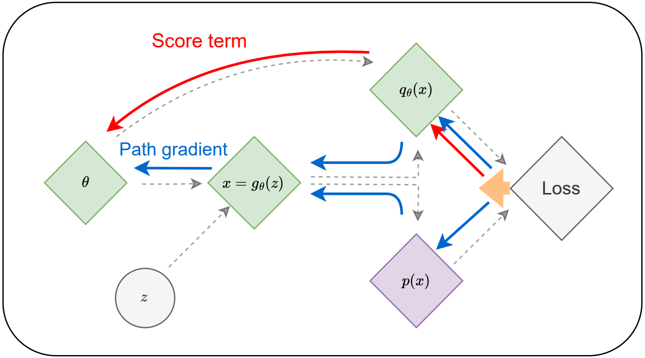
Most widely used *Real Information* *Noise*



- Score term has expectation 0, but non-vanishing variance

Total Gradient = **Path Gradient** + **Score Term**

Most widely used — *Real Information* — *Noise*

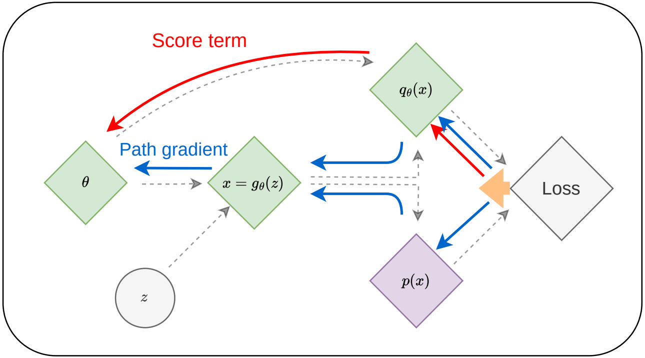


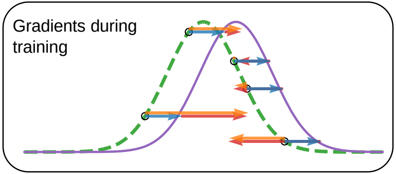
- Score term has expectation 0, but non-vanishing variance

  $\rightarrow$ path gradients are unbiased

Total Gradient = **Path Gradient** + **Score Term**

Most widely used *Real Information* *Noise*

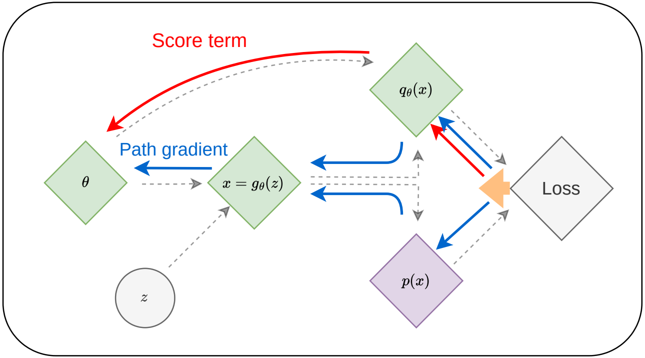


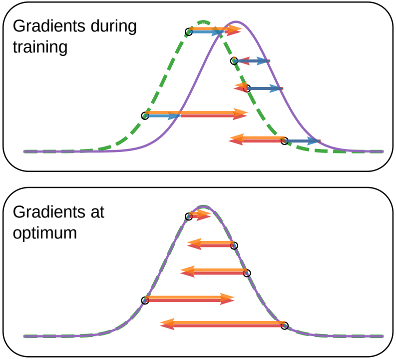


- Score term has expectation 0, but non-vanishing variance
  $\rightarrow$ path gradients are unbiased

Total Gradient = **Path Gradient** + **Score Term**

Most widely used *Real Information* *Noise*

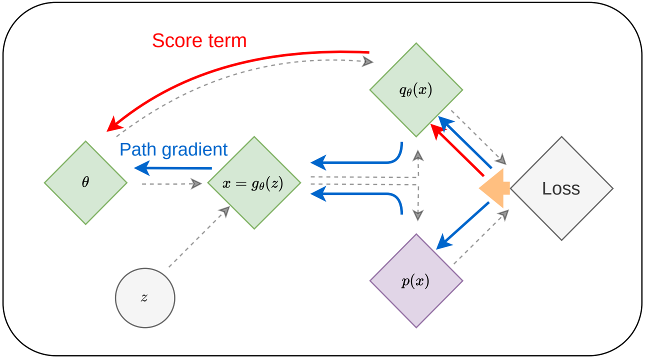


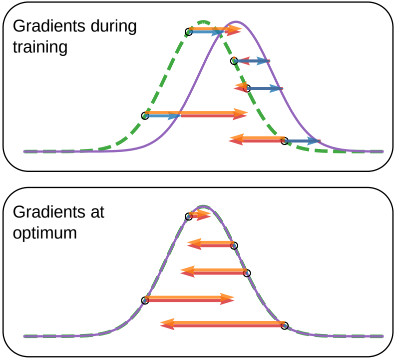


- Score term has expectation 0, but non-vanishing variance
  $\rightarrow$ path gradients are unbiased
- Path gradients are deterministically zero for $\dfrac{\partial p(x)}{\partial x} = \dfrac{\partial q_\theta(x)}{\partial x}$

Total Gradient = **Path Gradient** + **Score Term**

Most widely used — *Real Information* — *Noise*





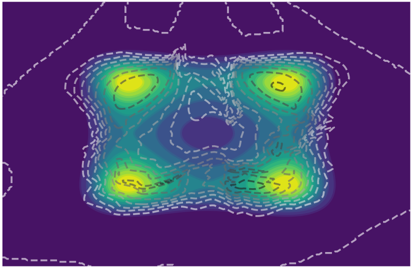
- Score term has expectation 0, but non-vanishing variance
  $\rightarrow$ path gradients are unbiased
- Path gradients are deterministically zero for $\frac{\partial p(x)}{\partial x} = \frac{\partial q_\theta(x)}{\partial x}$
- Favorable behavior observed by e.g. [Tucker et al., 2019, Geffner and Domke, 2021, Agrawal et al., 2020]

Path Gradients for the forward KL [Vaitl et al., 2024]

Path Gradients for the forward KL [Vaitl et al., 2024]

$$\mathbb{E}_{x\sim p}\left[\log p(x) \underbrace{- \log\det\left|\frac{\partial g_\theta^{-1}(x)}{\partial x}\right| - \log q_0(g_\theta^{-1}(x))}_{\log q_\theta}\right]$$

$$KL(p(x)|q_\theta(x))$$

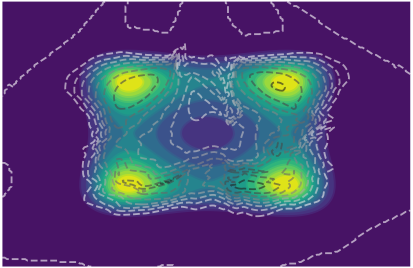


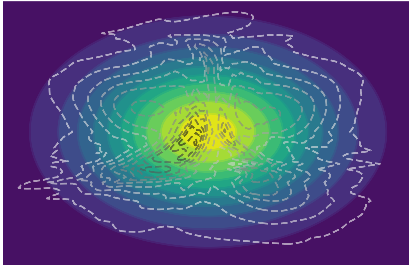
Path Gradients for the forward KL [Vaitl et al., 2024]

$$\mathbb{E}_{x\sim p}\Big[\overbrace{\log p(x) - \underbrace{\log\det\left|\frac{\partial g_\theta^{-1}(x)}{\partial x}\right|}^{\log p_{\theta,0}} - \log q_0(g_\theta^{-1}(x))}_{\log q_\theta}\Big]$$

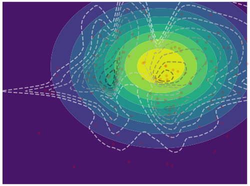
$$KL(p(x)|q_\theta(x)) \longleftrightarrow KL(p_{\theta,0}(z)|q_0(z))$$

duality of KL divergence



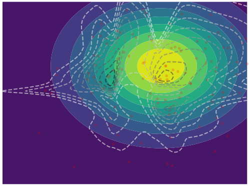


Maximum Likelihood



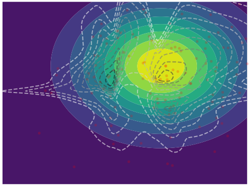
- Using the duality of KL divergence, easy to derive path gradients for forward KL

Maximum Likelihood

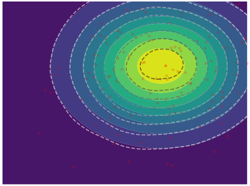


- Using the duality of KL divergence, easy to derive path gradients for forward KL
  - But need access to force of target $\frac{\partial \log p(x)}{\partial x}$

Maximum Likelihood

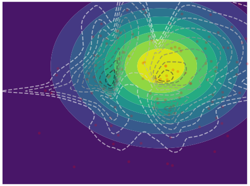


Path Gradients

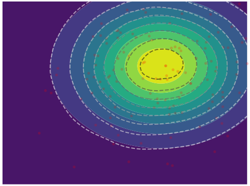


- Using the duality of KL divergence, easy to derive path gradients for forward KL
  - But need access to force of target $\frac{\partial \log p(x)}{\partial x}$

Maximum Likelihood

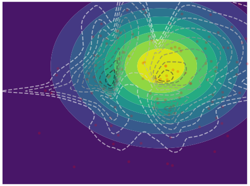


Path Gradients

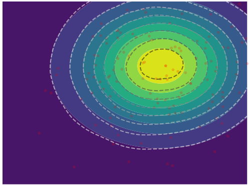


- Using the duality of KL divergence, easy to derive path gradients for forward KL
    - But need access to force of target $\frac{\partial \log p(x)}{\partial x}$
- We observed mitigating behavior of overfitting

Maximum Likelihood



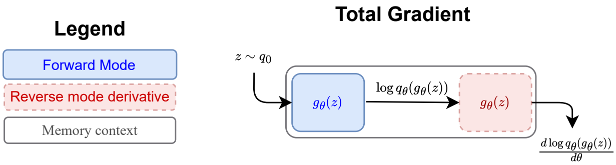
Path Gradients



- Using the duality of KL divergence, easy to derive path gradients for forward KL
  - But need access to force of target $\frac{\partial \log p(x)}{\partial x}$
- We observed mitigating behavior of overfitting
- We showed equivalence to GDReG estimator [Bauer and Mnih, 2021]

Computing Path gradients for Normalizing Flows

- While the total gradient is easy to compute, the path gradient is not



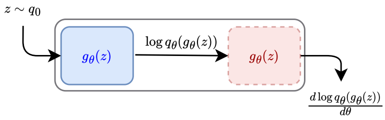
Computing Path gradients for Normalizing Flows

- While the total gradient is easy to compute, the path gradient is not
- Naive algorithm has twice the computations and memory cost

Legend

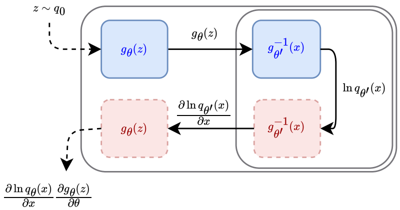
- Forward Mode
- Reverse mode derivative
- Memory context

Total Gradient



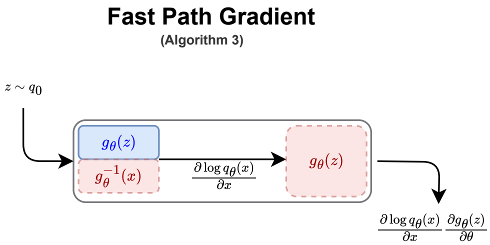
Naive Path Gradient

(Agrawal et al. 2020)



Fast Path gradients

- Idea: compute force term $\frac{\partial \log q_\theta(x)}{\partial x}$ along the sampling path

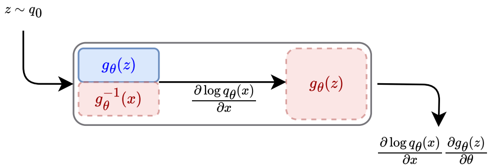


Fast Path gradients

- Idea: compute force term $\frac{\partial \log q_\theta(x)}{\partial x}$ along the sampling path
- No increase in memory cost

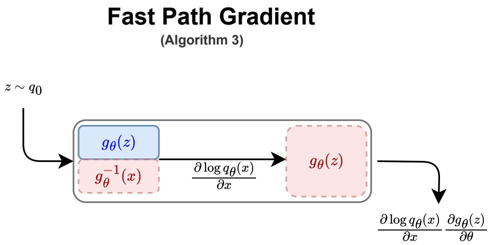
Fast Path Gradient

(Algorithm 3)



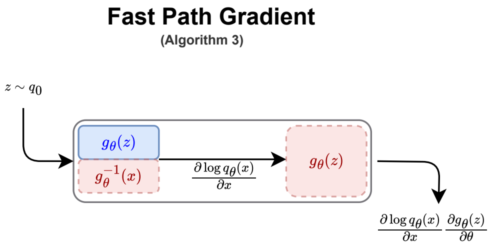
Fast Path gradients

- Idea: compute force term $\frac{\partial \log q_\theta(x)}{\partial x}$ along the sampling path
- No increase in memory cost
- Low increase in computational cost



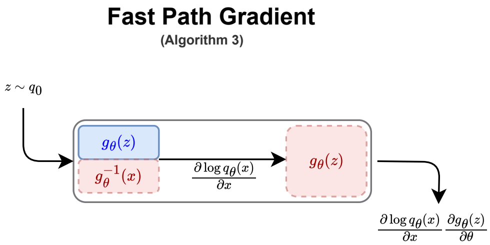
Fast Path gradients

- Idea: compute force term $\dfrac{\partial \log q_\theta(x)}{\partial x}$ along the sampling path
- No increase in memory cost
- Low increase in computational cost
- Derived for:



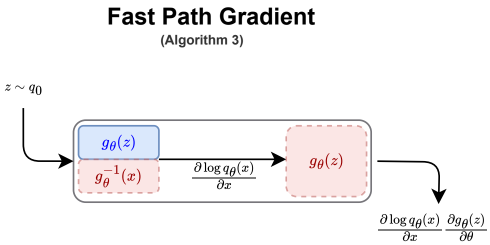
Fast Path gradients

- Idea: compute force term $\frac{\partial \log q_\theta(x)}{\partial x}$ along the sampling path
- No increase in memory cost
- Low increase in computational cost
- Derived for:
  - Continuous Normalizing Flows [Vaitl et al., 2022b]



Fast Path gradients

- Idea: compute force term $\dfrac{\partial \log q_\theta(x)}{\partial x}$ along the sampling path
- No increase in memory cost
- Low increase in computational cost
- Derived for:
  - Continuous Normalizing Flows [Vaitl et al., 2022b]
  - Coupling-type Normalizing Flows [Vaitl et al., 2024]



Continuous Normalizing Flows [Chen et al., 2018]

- Adjoint-state allows gradient computation with constant memory

$$
\begin{aligned}
x &= g_\theta(x_0) \\
&= x_0 + \int_0^T f_\theta(x_t, t)\,\mathrm{d}t \\
\log\left|\det \frac{\partial g_\theta}{\partial x_0}\right| &= \int_0^T \mathrm{tr}\left(\frac{\partial f_\theta(x_t, t)}{\partial x_t}\right)\mathrm{d}t
\end{aligned}
$$

Continuous Normalizing Flows [Chen et al., 2018]

- Adjoint-state allows gradient computation with constant memory
- Easy to incorporate symmetries [Köhler et al., 2020]

$$
\begin{aligned}
x &= g_\theta(x_0) \\
&= x_0 + \int_0^T f_\theta(x_t, t)\,\mathrm{d}t \\
\log\left|\det \frac{\partial g_\theta}{\partial x_0}\right| &= \int_0^T \mathrm{tr}\left(\frac{\partial f_\theta(x_t, t)}{\partial x_t}\right)\mathrm{d}t
\end{aligned}
$$

Generalized Adjoint State ODE [Vaitl et al., 2022b]

The derivative $\frac{\partial \log q_\theta(x_T)}{\partial x_T}$ can be obtained by solving the initial value problem

$$\begin{aligned} \frac{d}{dt}\frac{\partial \log q_\theta(x_t)}{\partial x_t} = &- \frac{\partial \log q_\theta(x_t)^\intercal}{\partial x_t}\frac{\partial f_\theta(x_t,t)}{\partial x_t} \\ &- \frac{\partial}{\partial x_t}\mathrm{Tr}\left(\frac{\partial f_\theta(x_t,t)}{\partial x_t}\right) \end{aligned}$$

Generalized Adjoint State ODE [Vaitl et al., 2022b]

The derivative $\frac{\partial \log q_\theta(x_T)}{\partial x_T}$ can be obtained by solving the initial value problem

$$\frac{d}{dt}\frac{\partial \log q_\theta(x_t)}{\partial x_t} = -\frac{\partial \log q_\theta(x_t)}{\partial x_t}^{\intercal}\frac{\partial f_\theta(x_t,t)}{\partial x_t} - \frac{\partial}{\partial x_t}\mathrm{Tr}\left(\frac{\partial f_\theta(x_t,t)}{\partial x_t}\right)$$

- Adjoint State ODE as in [Chen et al., 2018]

Generalized Adjoint State ODE [Vaitl et al., 2022b]

The derivative $\frac{\partial \log q_\theta(x_T)}{\partial x_T}$ can be obtained by solving the initial value problem

$$\frac{d}{dt}\frac{\partial \log q_\theta(x_t)}{\partial x_t} = -\frac{\partial \log q_\theta(x_t)}{\partial x_t}^\intercal \frac{\partial f_\theta(x_t,t)}{\partial x_t} - \frac{\partial}{\partial x_t}\mathrm{Tr}\left(\frac{\partial f_\theta(x_t,t)}{\partial x_t}\right)$$

- Additional term derived by us
- Adjoint State ODE as in [Chen et al., 2018]

Generalized Adjoint State ODE [Vaitl et al., 2022b]

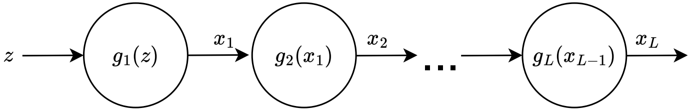
The derivative $\frac{\partial \log q_\theta(x_T)}{\partial x_T}$ can be obtained by solving the initial value problem

$$\frac{d}{dt}\frac{\partial \log q_\theta(x_t)}{\partial x_t} = -\frac{\partial \log q_\theta(x_t)}{\partial x_t}^\intercal \frac{\partial f_\theta(x_t, t)}{\partial x_t} - \frac{\partial}{\partial x_t}\mathrm{Tr}\left(\frac{\partial f_\theta(x_t, t)}{\partial x_t}\right)$$

- Additional term derived by us
- Adjoint State ODE as in [Chen et al., 2018]
- Constant memory and 33% runtime increase compared to total gradient

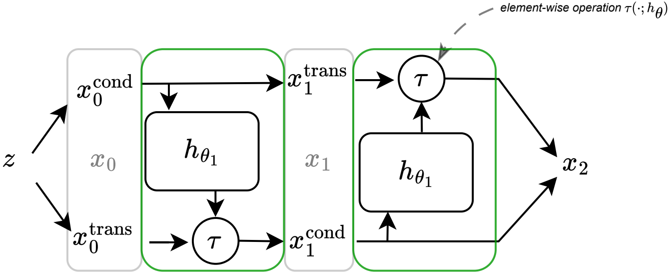
Compositional Flows

$$x = g_\theta(z) := g_{L,\theta_L} \circ \cdots \circ g_{1,\theta_1}(x_0),$$



$$\det \frac{\partial g_\theta(x_0)}{\partial x_0} = \prod_{l=1}^{L} \det \frac{\partial g_{l,\theta_l}(x_{l-1})}{\partial x_{l-1}}.$$

Coupling Flow



$$\frac{\partial g_1(x_0)}{\partial x_0} = \begin{pmatrix} I & 0 \\ \frac{\partial x^{\text{trans}}}{\partial x^{\text{cond}}} & D \end{pmatrix} \qquad \frac{\partial g_2(x_1)}{\partial x_1} = \begin{pmatrix} D & \frac{\partial x^{\text{trans}}}{\partial x^{\text{cond}}} \\ 0 & I \end{pmatrix}$$

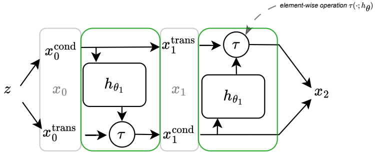
Recursive Gradient computation for coupling flows [Vaitl et al., 2024]

For a coupling flow the force $\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}}$ can be computed recursively with

$$\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x^{\text{trans}}_{l+1}} = \frac{\partial \log q_{\theta,l}(x_l)}{\partial x^{\text{trans}}_{l}} (D)^{-1} - \frac{\partial}{\partial x^{\text{trans}}_{l}} \log |\det D| \, (D)^{-1} \, ,$$

$$\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x^{\text{cond}}_{l+1}} = \frac{\partial \log q_{\theta,l}(x_l)}{\partial x^{\text{cond}}_{l}} - \frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x^{\text{trans}}_{l+1}} \frac{\partial \tau(x^{\text{trans}}_{l}; h_{\theta_{l+1}}(x^{\text{cond}}_{l}))}{\partial x^{\text{cond}}_{l}}$$

$$- \frac{\partial}{\partial x^{\text{cond}}_{l}} \log |\det D| \, ,$$



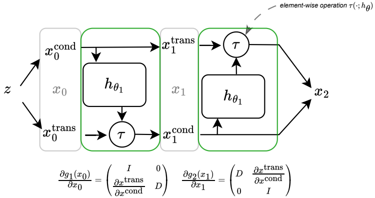
$$\frac{\partial g_1(x_0)}{\partial x_0} = \begin{pmatrix} I & 0 \\ \frac{\partial x^{\text{trans}}}{\partial x^{\text{cond}}} & D \end{pmatrix} \quad \frac{\partial g_2(x_1)}{\partial x_1} = \begin{pmatrix} D & \frac{\partial x^{\text{trans}}}{\partial x^{\text{cond}}} \\ 0 & I \end{pmatrix}$$

Recursive Gradient computation for coupling flows [Vaitl et al., 2024]

For a coupling flow the force $\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}}$ can be computed recursively with

$$\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}^{\text{trans}}} = \frac{\partial \log q_{\theta,l}(x_l)}{\partial x_l^{\text{trans}}} (D)^{-1} - \frac{\partial}{\partial x_l^{\text{trans}}} \log |\det D| \, (D)^{-1} \ ,$$

$$\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}^{\text{cond}}} = \frac{\partial \log q_{\theta,l}(x_l)}{\partial x_l^{\text{cond}}} - \frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}^{\text{trans}}} \frac{\partial \tau(x_l^{\text{trans}}; h_{\theta_{l+1}}(x_l^{\text{cond}}))}{\partial x_l^{\text{cond}}} - \frac{\partial}{\partial x_l^{\text{cond}}} \log |\det D| \ ,$$



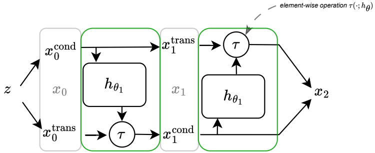
- Avoids expensive numerical inversion of τ

Recursive Gradient computation for coupling flows [Vaitl et al., 2024]

For a coupling flow the force $\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}}$ can be computed recursively with

$$\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}^{\text{trans}}} = \frac{\partial \log q_{\theta,l}(x_l)}{\partial x_l^{\text{trans}}}(D)^{-1} - \frac{\partial}{\partial x_l^{\text{trans}}} \log|\det D|\,(D)^{-1}\,,$$

$$\frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}^{\text{cond}}} = \frac{\partial \log q_{\theta,l}(x_l)}{\partial x_l^{\text{cond}}} - \frac{\partial \log q_{\theta,l+1}(x_{l+1})}{\partial x_{l+1}^{\text{trans}}}\frac{\partial \tau(x_l^{\text{trans}}; h_{\theta_{l+1}}(x_l^{\text{cond}}))}{\partial x_l^{\text{cond}}} - \frac{\partial}{\partial x_l^{\text{cond}}} \log|\det D|\,,$$



$$\frac{\partial g_1(x_0)}{\partial x_0} = \begin{pmatrix} I & 0 \\ \frac{\partial x^{\text{trans}}}{\partial x^{\text{cond}}} & D \end{pmatrix} \qquad \frac{\partial g_2(x_1)}{\partial x_1} = \begin{pmatrix} D & \frac{\partial x^{\text{trans}}}{\partial x^{\text{cond}}} \\ 0 & I \end{pmatrix}$$

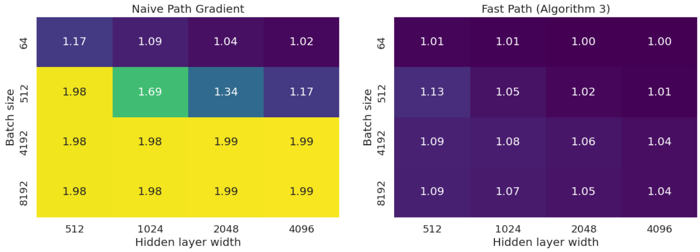
- Avoids expensive numerical inversion of τ
- Cheap inversion of diagonal matrix D

Benchmarks for algorithms

- Memory footprint

Benchmarks for algorithms

- Memory footprint



- Runtime increase

Coupling Type	Algorithm	Batch-size 8192
Affine	Recursive algorithm [Vaitl et al., 2024]	**1.4** $\pm$ **0.0**
	Baseline [Vaitl et al., 2022a]	2.1 $\pm$ 0.0
Non-invertible	Recursive algorithm [Vaitl et al., 2024]	**2.3** $\pm$ **0.0**
	Baseline using [Köhler et al., 2021]	8.2 $\pm$ 0.0

Results for Coupling Flows

- Machine Learning problem Multimodal Gaussian Mixture (MGM) with RealNVP

		Reverse KL	
		Total Gradient	Path Gradient
MGM	ESS_p	92.2 ± 0.0	**97.4** $\pm$ **0.0**
	ESS_q	93.0 ± 0.0	**97.4** $\pm$ **0.0**

Results for Coupling Flows

- Machine Learning problem Multimodal Gaussian Mixture (MGM) with RealNVP
- Lattice Field Theory problems:

		Reverse KL	
		Total Gradient	Path Gradient
MGM	ESS_p	92.2 ± 0.0	**97.4** $\pm$ **0.0**
	ESS_q	93.0 ± 0.0	**97.4** $\pm$ **0.0**

Results for Coupling Flows

- Machine Learning problem Multimodal Gaussian Mixture (MGM) with RealNVP
- Lattice Field Theory problems:
  - ϕ^4 theory with Z2Nice [Nicoli et al., 2021]

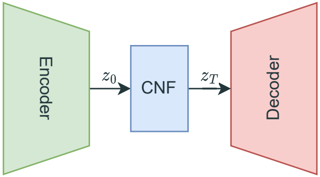
		Reverse KL	
		Total Gradient	Path Gradient
MGM	ESS_p	92.2 ± 0.0	**97.4** $\pm$ **0.0**
	ESS_q	93.0 ± 0.0	**97.4** $\pm$ **0.0**
ϕ^4	ESS_p	85.6 ± 0.1	**96.0** $\pm$ **0.1**
	ESS_q	85.6 ± 0.1	**96.0** $\pm$ **0.1**

Results for Coupling Flows

- Machine Learning problem Multimodal Gaussian Mixture (MGM) with RealNVP
- Lattice Field Theory problems:
  - ϕ^4 theory with Z2Nice [Nicoli et al., 2021]
  - $U(1)$ gauge theory with equivariant non-invertible coupling flow [Kanwar et al., 2020]

		Reverse KL	
		Total Gradient	Path Gradient
MGM	ESS_p	92.2 ± 0.0	**97.4** $\pm$ **0.0**
	ESS_q	93.0 ± 0.0	**97.4** $\pm$ **0.0**
ϕ^4	ESS_p	85.6 ± 0.1	**96.0** $\pm$ **0.1**
	ESS_q	85.6 ± 0.1	**96.0** $\pm$ **0.1**
$U(1)$	ESS_q	40.1 ± 0.0	**41.1** $\pm$ **0.0**
	ELBO	$1346.42 \pm .01$	**1346.43** $\pm$ **.00**

ML setting: VAE + FFJORD [Grathwohl et al., 2019]

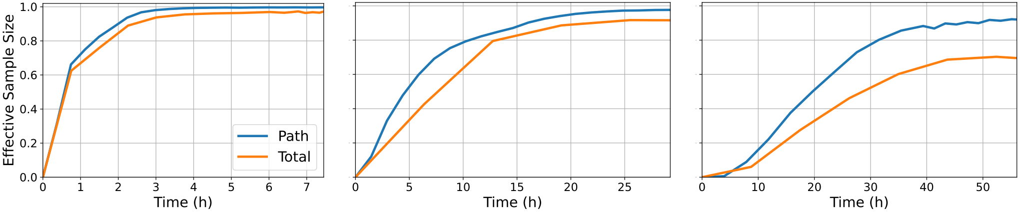


-ELBO	Path	Total
MNIST	**82.09** $\pm .04$	$82.82 \pm .01$
Omniglot	**96.61** $\pm .17$	$98.33 \pm .09$
Caltech Silhouettes	**101.93** $\pm .63$	$104.03 \pm .43$
Frey Faces	**4.35** $\pm .00$	$4.39 \pm .01$

Lattice Field Theory

ϕ^4 theory with equivariant CNF [de Haan et al., 2021]

Lattice size	Path	Total
12x12	**99.66**$\% \pm 0.07$	$98.01\% \pm 0.44$
20x20	**97.65**$\% \pm 0.14$	$91.56\% \pm 1.13$
32x32	**91.81**$\% \pm 1.32$	$69.53\% \pm 5.59$



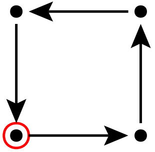
Lattice Gauge Theory

- 4D Yang-Mills Theory:

$$S = \frac{\beta}{3} \sum_{\mu,\nu<\mu,x} Re\mathrm{Tr}\left(I - P(x,\mu,\nu)\right)$$

  - $P(x,\mu,\nu)$: plaquette
  - β: inverse coupling
  - $P(x,\mu,\nu) \in SU(3)$

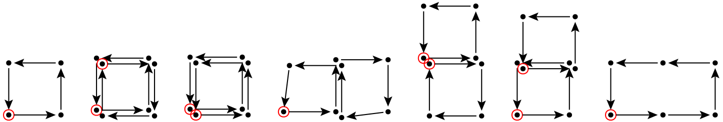
- Is a fundamental part of the Standard Model in Particle Physics
- YM models gluons, the force carriers of the strong force
- Has a huge group of local invariances



$P(\circ, 1, 2)$

Lüscher's approach [Lüscher, 2010]

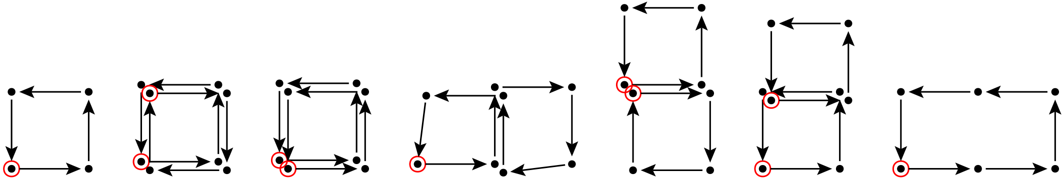
- Constructed a CNF by a Taylor Expansion
- Defined $\dot{U}(U_t, t, \theta) = \partial\tilde{S}(U_t, t) \cdot U_t$, where $\partial\tilde{S}(U_t, t)$ is force of generic action
  - $\tilde{S}$ is scalar & invariant
    - Force equivariant and is element of Lie algebra $\mathfrak{su}(N)$
    - generic ODE for lattice gauge theory
  - $\tilde{S} = \Sigma_i c_i(t) W_i(U_t)$
    - W_i are traces of Wilson loops
    - $c_i(t)$ are time dependent coefficients parametrized by θ
- Lüscher found $W_i \& c_i(t)$ by a perturbative expansion around $t = 0$



Training Trivializing Maps

- We proposed optimizing θ by minimizing the $\mathrm{KL}(q_\theta|p) \stackrel{c}{=} \mathbb{E}_{q_\theta(U_T)}\left[\ln q_\theta(U_T) + S(U_T)\right]$
  - Derived adjoint-state method for adjoint state $\in \mathfrak{su}(N)$
  - Implemented CG3 ODE solver, more complex functions for $c_i(t)$
  - Used **path gradients** for low variance gradient estimators

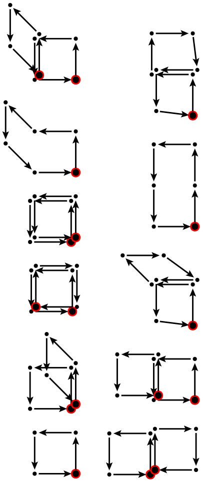
$\Rightarrow$ Expressive model with **few** parameters (14: linear $c_i(t)$ for each Wilson loop W_i)



Results Trivializing Map in 4D

Training

- 4D SU(3) Yang-Mills Theory
- 11 Wilson loops
- Target $\beta \in \{1, 2, 3\}$
  - $c_i(t)$ cubic splines with 2,5,7 knots
  - 5,10,15 ODE steps
- Lattice size 8, base-density uniform
- Batch-size 1, Adam, learning-rate 10^{-4}, trained on 1 A100
  - Trained on Juwels-Booster

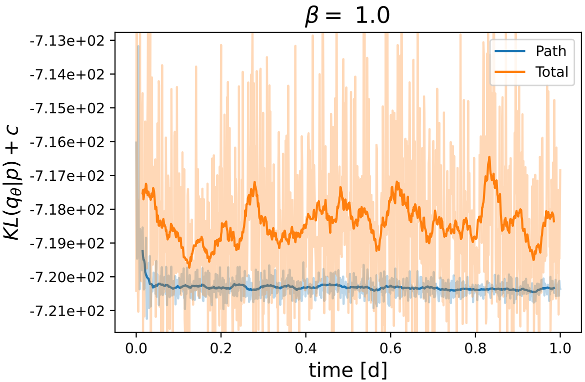


Results in 4D LGT

Baseline for $\beta = 1$ by [Abbott et al., 2023] is 75% Effective Sampling Ratio

Estimated on 1k samples

β	Path	Total	days trained
1	96.6 %	13.7 %	1

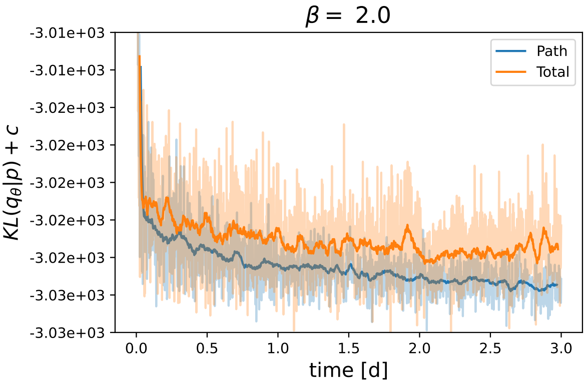


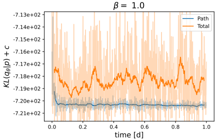
Results in 4D LGT

Baseline for $\beta = 1$ by [Abbott et al., 2023] is 75% Effective Sampling Ratio

Estimated on 1k samples

β	Path	Total	days trained
1	96.6 %	13.7 %	1
2	40.1 %	16.7 %	3



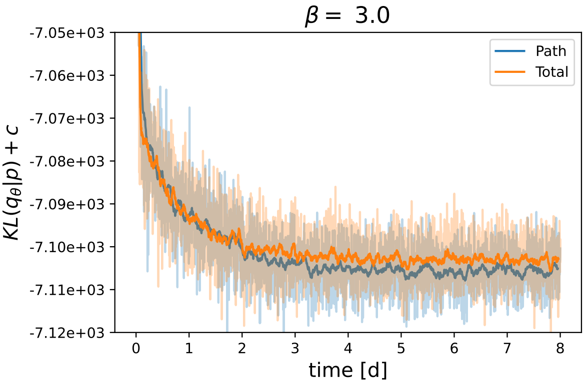


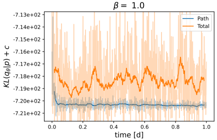
Results in 4D LGT

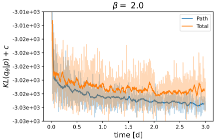
Baseline for $\beta = 1$ by [Abbott et al., 2023] is 75% Effective Sampling Ratio

Estimated on 1k samples

β	Path	Total	days trained
1	96.6 %	13.7 %	1
2	40.1 %	16.7 %	3
3	00.8 %	00.4 %	8

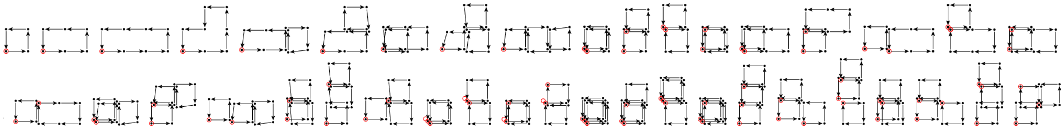






Summary

- Path gradients help training
- We expanded their applicability to continuous and coupling flows and training on target samples
- As is, the proposed CNF is not able to scale up to interesting β and lattice size
  - Possible to make flow more complex (e.g. NNLO basis), but drastic increase in runtime
  - Problem becomes exponentially more complex with increasing target β and lattice size



Thank you for your attention

References i

- Abbott, R., Albergo, M., Botev, A., Boyda, D., Cranmer, K., Hackett, D., Kanwar, G., Matthews, A., Racaniere, S., Razavi, A., et al. (2023).
  Normalizing flows for lattice gauge theory in arbitrary space-time dimension.
  ArXiv preprint, abs/2305.12345.
- Agrawal, A., Sheldon, D. R., and Domke, J. (2020).
  Advances in black-box VI: normalizing flows, importance weighting, and optimization.
  In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference*

on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.

Bauer, M. and Mnih, A. (2021).
Generalized doubly-reparameterized gradient estimators.
In *Third Symposium on Advances in Approximate Bayesian Inference*.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018).
Neural ordinary differential equations.
In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6572–6583.

References iii

- de Haan, P., Rainone, C., Cheng, M. C. N., and Bondesan, R. (2021).
  Scaling up machine learning for quantum field theory with equivariant continuous flows.
  CoRR, abs/2110.02673.
- Geffner, T. and Domke, J. (2021).
  Empirical evaluation of biased methods for alpha divergence minimization.
  arXiv preprint arXiv:2105.06587.

References iv

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., and Duvenaud, D. (2019).
Scalable reversible generative models with free-form continuous dynamics.
In *International Conference on Learning Representations*.

Kanwar, G., Albergo, M. S., Boyda, D., Cranmer, K., Hackett, D. C., Racaniere, S., Rezende, D. J., and Shanahan, P. E. (2020).
Equivariant flow-based sampling for lattice gauge theory.
Physical Review Letters, 125(12):121601.

Köhler, J., Klein, L., and Noé, F. (2020).
Equivariant flows: Exact likelihood generative learning for symmetric densities.
In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5361–5370. PMLR.

Köhler, J., Krämer, A., and Noé, F. (2021).
Smooth Normalizing Flows.
In *Advances in Neural Information Processing Systems 34*.

Lüscher, M. (2010).
Trivializing maps, the wilson flow and the hmc algorithm.
Communications in mathematical physics, 293:899–919.

Nicoli, K. A., Anders, C. J., Funcke, L., Hartung, T., Jansen, K., Kessel, P., Nakajima, S., and Stornati, P. (2021).
Estimation of thermodynamic observables in lattice field theories with deep generative models.
Phys. Rev. Lett., 126:032001.

Roeder, G., Wu, Y., and Duvenaud, D. (2017).
Sticking the landing: Simple, lower-variance gradient estimators for variational inference.
In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6925–6934.

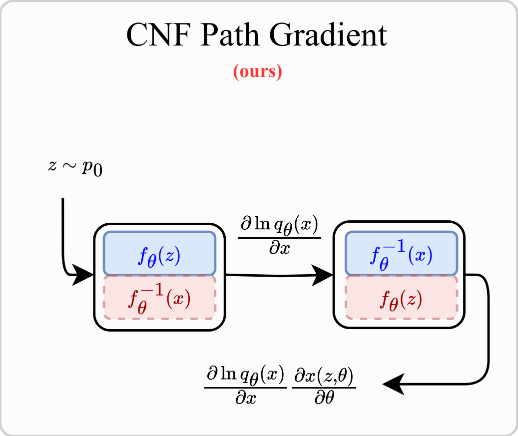
Tucker, G., Lawson, D., Gu, S., and Maddison, C. J. (2019).
Doubly reparameterized gradient estimators for monte carlo objectives.
In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

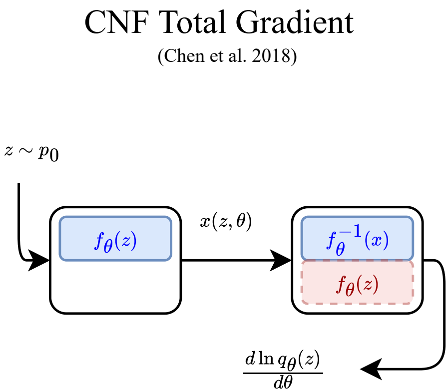
Vaitl, L., Nicoli, K. A., Nakajima, S., and Kessel, P. (2022a).
Gradients should stay on path: better estimators of the reverse-and forward kl divergence for normalizing flows.
Machine Learning: Science and Technology, 3(4):045006.

Vaitl, L., Nicoli, K. A., Nakajima, S., and Kessel, P. (2022b).
Path-gradient estimators for continuous normalizing flows.
In *International Conference on Machine Learning*, pages 21945–21959. PMLR.

Vaitl, L., Winkler, L., Richter, L., and Kessel, P. (2024).
Fast and unified path gradient estimators for normalizing flows.
In *to be presented in 12th International Conference on Learning Representations*.

Recursive path gradient algorithms CNF



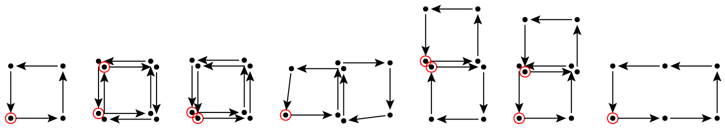


Reverse mode derivative

Forward mode

Lüscher's approach [Lüscher, 2010]

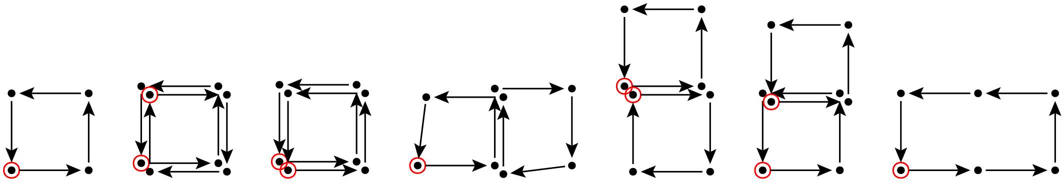
- Defined $\dot{U}(U_t, t, \theta) = \partial\tilde{S}(U_t, t) \cdot U_t$, where $\partial\tilde{S}(U_t, t)$ is force of generic action
  - $\tilde{S}$ is scalar & invariant
    - Force equivariant and is element of Lie algebra $\mathfrak{su}(N)$
    - generic ODE for lattice gauge theory
  - $\tilde{S} = \Sigma_i c_i(t) W_i(U_t)$
    - W_i are traces of Wilson loops
    - $c_i(t)$ are time dependent coefficients parametrized by θ
- Lüscher found $W_i \& c_i(t)$ by a perturbative expansion around $t = 0$



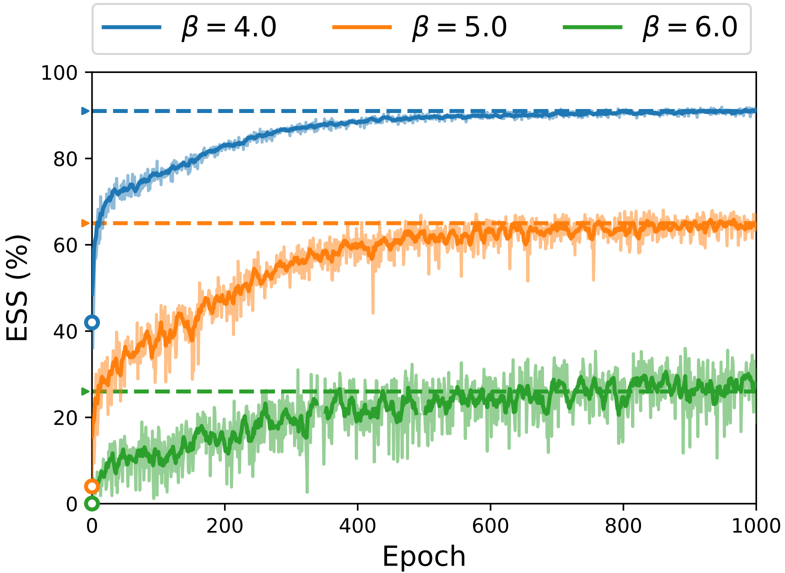
Training Trivializing Maps

- We proposed optimizing θ by minimizing the $\mathrm{KL}(q_\theta|p) \overset{c}{=} \mathbb{E}_{q_\theta(U_T)}\left[\ln q_\theta(U_T) + S(U_T)\right]$
  - Derived adjoint-state method for adjoint state $\in \mathfrak{su}(N)$
  - Implemented CG3 ODE solver, more complex functions for $c_i(t)$
  - Used **path gradients** for low variance gradient estimators

$\Rightarrow$ Expressive model with **few** parameters (14: linear $c_i(t)$ for each Wilson loop W_i)



2D SU(3) Yang-Mills Theory, $L = 16$



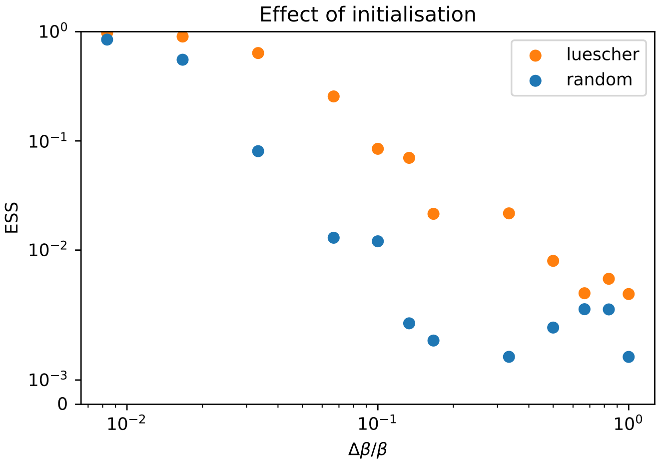
Results Trivializing Map in 4D

Training

- 4D SU(3) Yang-Mills Theory
- 11 Wilson loops
- Target $\beta \in \{1, 2, 3, 4\}$
  - $c_i(t)$ cubic splines with 2,5,7,10 knots
  - 5,10,15,20 ODE steps
- Lattice size 8, base-density uniform
- Batch-size 1, Adam, learning-rate 10^{-4}, trained on 1 A100
  - Trained on Juwels-Booster

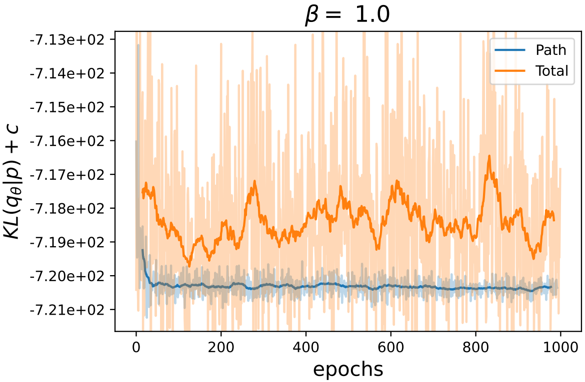
Training from non-trivial distribution

- 2D lattice, $L = 32$, target $\beta = 6$
- 1k epochs, batchsize 512



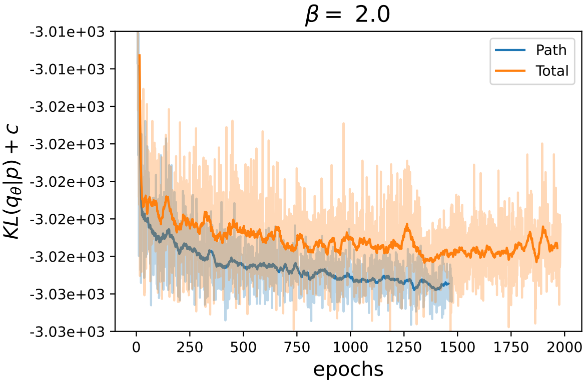
Results in 4D LGT

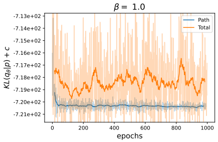
β	Path	Total	days trained
1	96.6 %	13.7 %	1



Results in 4D LGT

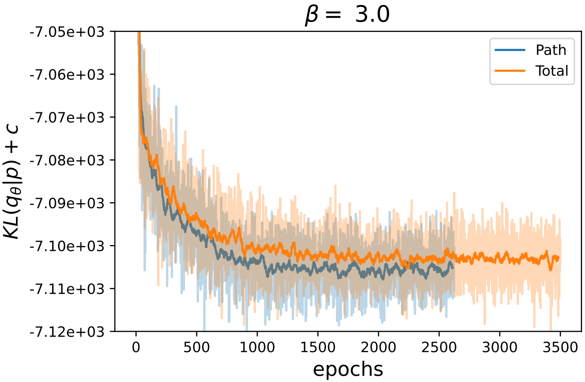
β	Path	Total	days trained
1	96.6 %	13.7 %	1
2	40.1 %	16.7 %	2

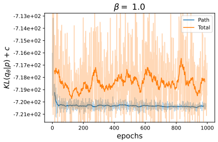


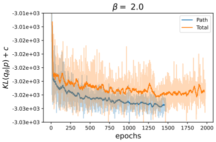


Results in 4D LGT

β	Path	Total	days trained
1	96.6 %	13.7 %	1
2	40.1 %	16.7 %	2
3	00.8 %	00.4 %	8

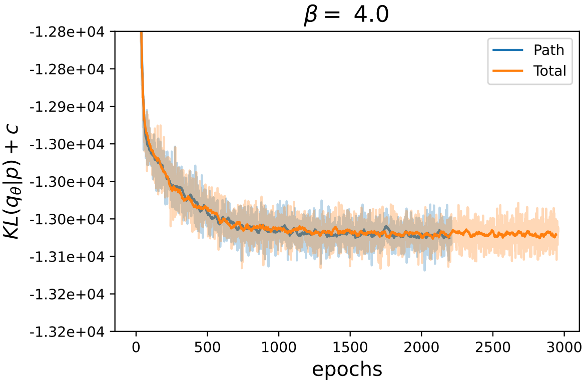


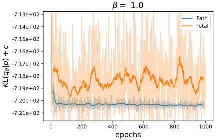


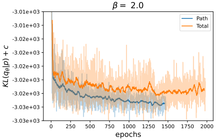


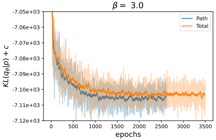
Results in 4D LGT

β	Path	Total	days trained
1	96.6 %	13.7 %	1
2	40.1 %	16.7 %	2
3	00.8 %	00.4 %	8
4	00.2 %	00.1 %	9









Acceptance rate, 4D LGT

Acceptance rate

Estimated on 1k samples

β	Path	Total	days trained
1	91 %	28 %	1
2	49 %	25 %	3
3	1 %	1 %	8
4	0 %	0 %	9

2D coefficients