Supplementary Material for "Better Training using Weight-Constrained Stochastic Dynamics"

Overview of the provided supplementary material:

Appendix A: Provides the results necessary to establish exponential convergence to equilibrium of constrained overdamped Langevin dynamics (M-7).

Appendix B: Provides discretization schemes and implementation details for our constrained training algorithms. The discretization schemes for a general constraint are described in Appendix B.1 for overdamped Langevin dynamics and in B.2 for underdamped Langevin dynamics. Our c-CoLA circle constrained algorithm is discussed in Appendix B.3 (overdamped) and B.4 (underdamped). Appendix B.5 and B.6 are reserved for our o-CoLA, orthogonality constraint Langevin dynamics, algorithm (overdamped and underdamped, respectively).

Appendix C: Illustrates the connection between the magnitude of the weights, the vanishing/exploding gradient problem, and the smoothness of the interpolant.

Appendix D: Provides further numerical details and additional results for our constrained methods.

Notation: the use of (M-...) in references refers to equations in the main paper.

A. Theory of constrained overdamped Langevin dynamics

We present here the details of the theory summarized in Sec. 4. In particular, we provide the key results to establish the exponential convergence to equilibrium of constrained overdamped Langevin dynamics Eq. (M-7).

In the first part (Sec. A.1), we derive the underlying SDE associated with Eq. (M-7), its generator and the invariant measure ν_Σ defined as

$$\mathrm{d}\nu_\Sigma = Z^{-1} e^{-\beta V(q)}\,\mathrm{d}\sigma_\Sigma, \quad Z = \int_\Sigma e^{-\beta V(q)}\,\mathrm{d}\sigma_\Sigma, \quad (1)$$

where σ_Σ is the surface measure on Σ. Ergodicity ensures that averages of observables with respect to ν_Σ can be approximated by time averages of trajectories of Eq. (M-7): for all test function $\phi \in \mathcal{C}_c^\infty(\Sigma)$

$$\lim_{T\to\infty} \langle\phi\rangle_T = \langle\phi\rangle_{\nu_\Sigma} \quad \text{for a.e. } q_0 \in \Sigma,$$
$$\langle\phi\rangle_T := \frac{1}{T}\int_0^T \phi(q_t)\,\mathrm{d}t, \ \langle\phi\rangle_{\nu_\Sigma} := \int_\Sigma \phi(q)\,\mathrm{d}\nu_\Sigma(q). \quad (2)$$

Next, in Sec. A.2 we present the Poincaré inequality on a manifold, which holds under a curvature-dimension assumption: there exists $\rho > 0$ such that

$$CD(\rho,\infty): \qquad \mathrm{Ric}_{\mathfrak{g}} + \beta\nabla^2_{\mathfrak{g}} V \geq \rho\mathfrak{g}, \quad (3)$$

in the sense of symmetric matrices, where $\mathfrak{g}$ is the Riemannian metric, $\mathrm{Ric}_{\mathfrak{g}}$ is the Ricci curvature tensor and $\nabla^2_{\mathfrak{g}} V$ is the Hessian of V on the manifold. Under Eq. (3) the following result holds.

Theorem A.1. *Assume that there exists $\rho > 0$ and $N > n$ such that $CD(\rho, N)$ holds. Then ν_Σ satisfies a Poincaré inequality: there exists a constant $L > 0$ such that*

$$\int_\Sigma \left|\phi(q) - \langle\phi\rangle_{\nu_\Sigma}\right|^2 \mathrm{d}\nu_\Sigma(q) \leq \frac{1}{2L}\int_\Sigma \left|\Pi(q)\nabla\phi(q)\right|^2 \mathrm{d}\nu_\Sigma(q)$$
$$\forall \phi \in H^1(\nu_\Sigma), \quad (4)$$

where $\Pi(q)$ is the projection onto the cotangent space $T^*_q\Sigma$ Eq. (8) and $H^1(\nu_\Sigma)$ is the space of functions with square ν_Σ-integrable gradients Eq. (7).

Consequences of Theorem A.1 are the exponential convergence and a central limit theorem (CLT) for the convergence in Eq. (2)

Corollary A.2. *If Eq. (3) holds then*

$$\int_\Sigma \left|\mathbb{E}(\phi(q_t) \mid q_0) - \langle\phi\rangle_{\nu_\Sigma}\right|^2 \mathrm{d}\nu_\Sigma(q_0) \leq C(\phi)e^{-2L/\beta t}$$
$$\forall \phi \in H^1(\nu_\Sigma), \quad (5)$$

where $C(\phi)$ depends only on ϕ. Furthermore we have the following convergence in law:

$$\sqrt{T}\big(\langle\phi\rangle_T - \langle\phi\rangle_{\nu_\Sigma}\big) \to \mathcal{N}(0,\sigma^2_\phi) \quad \text{as } T\to\infty,$$

where the asymptotic variance σ^2_ϕ is bounded as $\sigma^2_\phi \leq \frac{\beta}{L}\int_\Sigma \left|\phi - \langle\phi\rangle_{\nu_\Sigma}\right|^2 \mathrm{d}\nu_\Sigma$.

Appx. A.3 is dedicated to using the Poincaré inequality to proving this.

Notation

We collect here additional notation needed for this discussion.

Given a measure μ in a space $E \subset \mathbb{R}^d$, we associate the space of square integrable functions

$$L^2(\mu) = \big\{\phi : E \to \mathbb{R} \text{ measurable } : \int_E |\phi|^2\,\mathrm{d}\mu < \infty\big\}.$$

Equipped with the inner product and associated norm

$$\langle\phi,\psi\rangle_\mu = \int_E \phi\psi\,\mathrm{d}\mu, \qquad \|\phi\|_{L^2(\mu)} = \sqrt{\langle\phi,\phi\rangle},$$

$L^2(\mu)$ is a Hilbert space. We further define the subspace $L^2(\mu)$ of functions with zero mean by

$$L_0^2(\mu) = \big\{\phi \in L^2(\mu) : \langle\phi\rangle_\mu = 0\big\}, \quad \langle\phi\rangle_\mu = \int_E \phi \mathrm{d}\mu, \tag{6}$$

as well as the space of functions with square integrable gradient

$$H^1(\mu) = \big\{\phi \in L^2(\mu) : \partial_i\phi \in L^2(\mu) \quad 1 \le i \le d\big\}. \tag{7}$$

For the constraint $g : \mathbb{R}^d \to \mathbb{R}^m$, we denote the Jacobian matrix as $G(q) = \nabla_q^T g(q)$ and denote its right pseudo-inverse by $G^+ = G^T(GG^T)^{-1}$ (GG^T is invertible if G has full row rank). We verify that the map

$$\Pi : \mathbb{R}^d \to \mathbb{R}^{d\times d}, \quad q \mapsto \Pi(q) = I_d - G^+(q)G(q), \tag{8}$$

defines for each q the orthogonal projection onto the cotangent space $T_q^*\Sigma$.

$$\Pi_q = \Pi(q) : \mathbb{R}^d \to \mathbb{R}^d, \quad p \mapsto \Pi(q)p.$$

In particular, for all q we have $\Pi_q p \in T_q^*\Sigma$ and the matrix Π_q is symmetric and idempotent: (i.e., $\Pi_q^T = \Pi_q$ and $\Pi_q^2 = \Pi_q$).

A.1. The underlying SDE and the invariant measure

Although presented differently, the results of this section follow closely the treatment of this issue presented in (Lelièvre et al., 2010).

We define the mean curvature of the manifold as the vector valued function

$$\mathcal{H} : \mathbb{R}^d \to \mathbb{R}^d, \quad q \mapsto \big(\mathcal{H}(q)\big)_i = \Pi_{jk}(q)\partial_j\Pi_{ik}(q) \quad 1 \le i \le d, \tag{9}$$

where $\Pi(q) : \mathbb{R}^d \to \mathbb{R}^d$ is the projection onto the cotangent space defined in Eq. (8). We then establish the following result (proved below).

Lemma A.3. *The constrained system Eq. (M-7) can be rewritten as the following SDE in* $\mathbb{R}^d$

$$\begin{aligned}\mathrm{d}q_t = &-\Pi(q_t)\nabla V(q_t)\mathrm{d}t + \sqrt{2\beta^{-1}}\,\Pi(q_t)\mathrm{d}\mathcal{W}_t \\ &+ \beta^{-1}\mathcal{H}(q_t)\,\mathrm{d}t.\end{aligned} \tag{10}$$

The uniqueness of the invariant measure of Eq. (10) and the resulting ergodicity result Eq. (2) are proved in (Lelièvre et al., 2010)[Prop. 3.20] (the proof relies on the divergence theorem on manifolds).

The generator associated with Eq. (10) is given by

$$\mathcal{L} = -\Pi(q)\nabla V(q)\cdot\nabla + \beta^{-1}\mathcal{H}(q)\cdot\nabla + \beta^{-1}\Pi(q) : \nabla^2.$$

We verify that $\mathcal{L}$ can be written in the following symmetric form

$$\begin{aligned}\mathcal{L}\psi &= \beta^{-1}\operatorname{div}_\Sigma(\nabla_\Sigma\psi) - \nabla_\Sigma V(q)\cdot\nabla_\Sigma\psi \\ &= \beta^{-1}e^{\beta V(q)}\operatorname{div}_\Sigma\big(e^{-\beta V(q)}\nabla_\Sigma\psi\big),\end{aligned} \tag{11}$$

where we denote $\nabla_\Sigma\phi = \Pi\nabla\phi$ and $\operatorname{div}_\Sigma\psi = \nabla_\Sigma\cdot\psi = \sum_{i,j=1}^d \Pi_{ij}\partial_j\psi_i$. This expression directly implies that $\mathcal{L}$ is reversible with respect to ν_Σ:

$$\big\langle\mathcal{L}\phi,\psi\big\rangle_{\nu_\Sigma} = -\beta^{-1}\big\langle\nabla_\Sigma\phi,\nabla_\Sigma\psi\big\rangle_{\nu_\Sigma} = \big\langle\phi,\mathcal{L}\psi\big\rangle_{\nu_\Sigma}. \tag{12}$$

Thanks to this expression, we can prove that the measure ν_Σ is indeed invariant for Eq.(M-7). Let us introduce the forward Kolmogorov equation: given a test function $\phi \in \mathcal{C}_c^\infty(\Sigma)$

$$\partial_t u(t,q) = \mathcal{L}u(t,q)\ t \ge 0,\ q \in \Sigma \qquad u(0,q) = \phi(q).$$

The solution to this equation is verified to be $u(t,q) = \mathbb{E}(\phi(q_t) \mid q_0 = q)$ (see the Feynmann–Kac formula) and is usually denoted as $u(t,q) = e^{t\mathcal{L}}\phi(q)$. The measure ν_Σ is invariant if for any $t \ge 0$ $\int_\Sigma u(t,q)\,\mathrm{d}\nu_\Sigma(q) = \int_\Sigma u(0,q)\,\mathrm{d}\nu_\Sigma(q) = \langle\phi\rangle_{\nu_\Sigma}$. This is easily verified thanks to Eq. (12):

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\int_\Sigma u(t,q)\,\mathrm{d}\nu_\Sigma(q) = \frac{\mathrm{d}}{\mathrm{d}t}\int_\Sigma e^{t\mathcal{L}}\phi(q)\,\mathrm{d}\nu_\Sigma(q) \\ &= \int_\Sigma \mathcal{L}e^{t\mathcal{L}}\phi(q)\,\mathrm{d}\nu_\Sigma(q) = \big\langle\mathcal{L}e^{t\mathcal{L}}\phi,\mathbf{1}\big\rangle_{\nu_\Sigma} = 0.\end{aligned}$$

Proof. Let us write λ_t as the Itô process

$$\mathrm{d}\lambda_t = \mu(q_t)\,\mathrm{d}t + \sigma(q_t)\,\mathrm{d}\mathcal{W}_t, \tag{13}$$

where $\mu : \mathbb{R}^d \to \mathbb{R}^m$, $\sigma : \mathbb{R}^d \to \mathbb{R}^{m\times d}$ and $\mathcal{W}_t$ is the same Wiener process as in Eq. (M-7). Using this expression in Eq. (M-7) brings

$$\begin{aligned}\mathrm{d}q_t = &\big(-\nabla V(q_t) - G(q_t)^T\mu(q_t)\big)\,\mathrm{d}t \\ &+ \big(\sqrt{2\beta^{-1}}I - G(q_t)^T\sigma(q_t)\big)\,\mathrm{d}\mathcal{W}_t,\end{aligned}$$

where we recall the notation for the Jacobian $G = \nabla_q^T g$. Using Itô formula we find

$$\begin{aligned}0 = \mathrm{d}g(q_t) &= G(q_t)\,\mathrm{d}q + b_t\,\mathrm{d}t \\ &= G(q_t)\big(-\nabla V(q_t)\,\mathrm{d}t + \sqrt{2\beta^{-1}}\,\mathrm{d}\mathcal{W}_t - G(q_t)^T\mathrm{d}\lambda_t\big) \\ &\quad + b_t\,\mathrm{d}t,\end{aligned} \tag{14}$$

where b_t is the d-dimensional process defined as (omitting the dependence on q_t)

$$\begin{aligned}(b_t)_i &= \frac{1}{2}\big(\sqrt{2\beta^{-1}}I - G^T\sigma\big)\big(\sqrt{2\beta^{-1}}I - G^T\sigma\big)^T : \nabla^2 g_i \\ &= \beta^{-1}\Delta g_i - \frac{\sqrt{2\beta^{-1}}}{2}\big(G^T\sigma + \sigma^T G\big) : \nabla^2 g_i \\ &\quad + \frac{1}{2}G^T\sigma\sigma^T G : \nabla^2 g_i.\end{aligned} \tag{15}$$

From Eq. (14) yields

$$\mathrm{d}\lambda_t = \big(G(q_t)G(q_t)^T\big)^{-1}G(q_t)\Big(- \nabla V(q_t)\,\mathrm{d}t + \sqrt{2\beta^{-1}}\,\mathrm{d}\mathcal{W}_t\Big) + \big(G(q_t)G(q_t)^T\big)^{-1}b_t\,\mathrm{d}t. \quad (16)$$

Identifying with Eq. (13) we find $\sigma(q) = \sqrt{2\beta^{-1}}(G^+(q))^T$, which used in Eq. (15) yields

$$(b_t)_i = \beta^{-1}\big(\Delta g_i - \big(G^T(G^+)^T + G^+G\big) : \nabla^2 g_i + G^T(G^+)^T G^+G : \nabla^2 g_i\big).$$

As G^+G is symmetric and $GG^+ = I_m$, we obtain

$$(b_t)_i = \beta^{-1}\big(\Delta g_i - G^+G : \nabla^2 g_i\big) = \beta^{-1}\Pi : \nabla^2 g_i. \quad (17)$$

Inserting Eq. (16) in Eq. (M-7) brings

$$\mathrm{d}q_t = -\Pi(q_t)\nabla V(q_t)\mathrm{d}t + \sqrt{2\beta^{-1}}\,\Pi(q_t)\mathrm{d}\mathcal{W}_t - G^+(q_t)b_t\,\mathrm{d}t. \quad (18)$$

To conclude the proof we require the following technical relations on the mean curvature vector (Eq. (19a) follows from a direct computation; the proof of Eq. (19b) is direct but involved and can be found in (Lelièvre et al., 2010))

Lemma A.4. *The projection Π and the vector H defined in Eq. (8) and Eq. (9) satisfy the following equalities*

$$\mathcal{H} = (I - \Pi)\nabla \cdot \Pi, \quad (19a)$$

$$\Pi : \nabla^2 g_i = -(G\mathcal{H})_i \qquad 1 \le i \le d, \quad (19b)$$

Equality Eq. (19a) ensures that $\Pi\mathcal{H} = 0$. Combining Eq. (17) and Eq. (19b) we can write $b_t = -\beta^{-1}G\mathcal{H}$. Thanks to these relations and the definition of Π, we obtain

$$-G^+b_t = \beta^{-1}G^+G\mathcal{H} = \beta^{-1}(I - \Pi)\mathcal{H} = \beta^{-1}\mathcal{H}.$$

This equality combined with Eq. (18) proves Eq. (A.3) and concludes the proof of Lemma A.3.

□

A.2. Poincaré inequality on a manifold

Poincaré inequalities, also called spectral gap inequalities, form an important family of functional inequalities in the theory of Markov diffusion processes. They are the simplest inequalities that provide results on the convergence to equilibrium. Stronger results can be obtained with the family of log-Sobolev inequalities, which are at the center of the Bakry–Émery theory (Bakry & Émery, 1985). We follow here closely the book (Bakry et al., 2013) on this subject (more specifically §1.16.2 and sections 4.2, 4.8, C.6). For the necessary terminology of Riemannian manifolds we recommend the introductory textbook (Lee, 2018) (the literature on this topic is vast and contains many works of high quality).

As presented in (Bakry et al., 2013)[Chap. 4], a Poincaré inequality can be obtained as a consequence of a curvature-dimension condition. For the sake of presentation, we introduce this result in the setting of a weighted Riemannian manifold. Let $(\mathcal{M}, \mathfrak{g})$ be an n-dimensional Riemannian manifold, where $\mathfrak{g}$ is the Riemannian metric. We consider the diffusion operator

$$\mathcal{L} = \Delta_{\mathfrak{g}} - \langle \nabla_{\mathfrak{g}} W, \nabla_{\mathfrak{g}} \cdot \rangle_{\mathfrak{g}},$$

where $\Delta_{\mathfrak{g}}$ denotes the Laplace–Beltrami operator on the manifold $\mathcal{M}$, $\nabla_{\mathfrak{g}}$ denotes the Levi–Civita connection (covariant derivative) and $\langle \cdot, \cdot \rangle_{\mathfrak{g}}$ denotes the Riemannian metric ($\langle X, Y \rangle_{\mathfrak{g}} = \mathfrak{g}(X, Y)$ for all vector fields X, Y). We verify that the associated invariant measure is $\mathrm{d}\mu = Z^{-1}e^{-W}\mathrm{d}\mu_{\mathfrak{g}}$, where $\mathrm{d}\mu_{\mathfrak{g}}$ is the Riemannian measure (Bakry et al., 2013)[§1.11.3]. For $N \in [n, \infty]$, we define the 2-tensor

$$\mathrm{Ric}_N(\mathcal{L}) = \mathrm{Ric}_{\mathfrak{g}} + \nabla^2_{\mathfrak{g}} W - \frac{1}{N-n}\mathrm{d}W \otimes \mathrm{d}W.$$

where $\mathrm{Ric}_{\mathfrak{g}}$ is the Ricci curvature 2-tensor and $\nabla^2_{\mathfrak{g}}$ denotes the Hessian operator on $\mathcal{M}$ (the case $N = n$ is considered only if W is constant). In this context, a curvature-dimension condition $CD(\rho, N)$ for $\rho \in \mathbb{R}$ and $N \ge n$ holds if and only if (see (Bakry et al., 2013)[C.6])

$$CD(\rho, N): \qquad \mathrm{Ric}_N(\mathcal{L}) \ge \rho\mathfrak{g}, \quad (20)$$

in the sense of symmetric $(0, 2)$-tensors (covariant 2-tensors). In the flat space $\mathcal{M} = \mathbb{R}^n$, the condition $CD(\rho, \infty)$ reads $\nabla^2 W \ge \rho I$, which is nothing but the convexity of the potential W. Under $CD(\rho, N)$, the measure μ is proved to satisfy a Poincaré inequality (in (Bakry et al., 2013), combine Thm 4.8.4 with the discussion in section C.6).

Theorem A.5. *(Bakry et al., 2013)[Thm 4.8.4] Under the curvature-dimension condition $CD(\rho, N)$ with $\rho > 0$ and $N \ge n$, $N > 1$, the measure μ satisfies the Poincaré inequality*

$$\mathrm{Var}_\mu(\phi) = \|\phi - \langle\phi\rangle_\mu\|^2_{L^2(\mu)} \le C_P\|\nabla_{\mathfrak{g}}\phi\|^2_{L^2(\mu)} \quad (21)$$

with constant $C_P = \frac{N-1}{\rho N}$, $\forall \phi \in L^2(\mu) \cap H^1(\mu)$.

As the tensor $\mathrm{d}W \otimes \mathrm{d}W$ is positive semi-definite, we verify the monotonicity $\mathrm{Ric}_{N+M}(\mathcal{L}) \ge \mathrm{Ric}_N(\mathcal{L})$ for any $M \ge 0$. This implies in particular that $CD(\rho, N) \Rightarrow CD(\rho, \infty)$ for any $N \in [n, \infty]$. Hence, among all choices of $N \ge n$, $CD(\rho, \infty)$ is the weaker condition.

Let us now consider this result in the context of the constraint manifold Σ in Eq. (M-1). We consider the space $\mathbb{R}^d$ with its Riemannian manifold structure given by the Euclidean metric $\bar{\mathfrak{g}}(v, w) = v \cdot w$ for all $v, w \in \mathbb{R}^d$ (for all

$q \in \mathbb{R}^d$, $p \in T_q\mathbb{R}^d$ is identified with $\mathbb{R}^d$ through a canonical isomorphism). Assuming that g is smooth and that $\nabla_q^T g$ has everywhere full row-rank, Σ is a smooth embedded submanifold of $\mathbb{R}^d$ of dimension $n = d - m$ (see e.g. (Lee, 2018)). Furthermore, Σ is equipped with the metric induced by $\bar{\mathfrak{g}}$: for a local parameterization of $\psi : U \subset \Sigma \to \mathbb{R}^d$, $\bar{\mathfrak{g}}$ is given locally on U by

$$\bar{\mathfrak{g}} = \sum_{i=1}^{d} \sum_{j,k=1}^{n} \frac{\partial \psi^i}{\partial x^j} \frac{\partial \psi^i}{\partial x^k} \mathrm{d}x^j \mathrm{d}x^k = \left(\nabla_x \psi \nabla_x^T \psi\right)_{jk} \mathrm{d}x^j \mathrm{d}x^k \tag{22}$$

We now define the potential $W = \beta V|_\Sigma$, where $V|_\Sigma$ denotes the restriction of V to Σ. Assumption 3 corresponds then to condition $CD(\rho, \infty)$ above. Applying Theorem A.5 we obtain Poincaré's inequality on the constraint manifold Σ. We note that for a function ϕ defined on $\mathbb{R}^d$, the covariant derivative in $\mathbb{R}^d$ of $\phi|_\Sigma$ on the manifold is the orthogonal projection of the directional derivative of ϕ (in the ambient manifold $\mathbb{R}^d$) onto the cotangent space: $\nabla_{\mathfrak{g}}(\phi|_\Sigma)(q) = \Pi(q)\nabla_q \phi(q)$. Furthermore, we note that the surface measure σ_Σ equals the Riemannian measure on the manifold (compare (Lelièvre et al., 2010)[Rem. 3.4] with (Lee, 2018)[Prop. 2.41] and Eq. (22)). We thus obtain the result of Theorem A.1 with constant $C_P = \frac{1}{\rho} = \frac{1}{2L}$.

A.3. Exponential convergence to equilibrium and central limit theorem

Let us define the norm of a linear operator $\mathcal{A} : L_0^2(\nu_\Sigma) \to L_0^2(\nu_\Sigma)$ as

$$\|\mathcal{A}\|_{\mathcal{B}(L_0^2(\nu_\Sigma))} = \sup_{\phi \in L_0^2(\nu_\Sigma)} \frac{\|\mathcal{A}\phi\|_{L_0^2(\nu_\Sigma)}}{\|\phi\|_{L_0^2(\nu_\Sigma)}}.$$

Denote $\bar{\phi} = \phi - \langle \phi \rangle_{\nu_\Sigma} \in L_0^2(\nu_\Sigma)$. The Poincaré inequality Eq. (4), rewritten on the subspace $L_0^2(\nu_\Sigma)$, is as follows:

$$\|\bar{\phi}\|^2_{L_0^2(\nu_\Sigma)} \leq \frac{1}{2L} \|\nabla_\Sigma \bar{\phi}\|^2_{L_0^2(\nu_\Sigma)} \; \forall \bar{\phi} \in L_0^2(\nu_\Sigma) \cap H^1(\nu_\Sigma). \tag{23}$$

Using the reversibility of the measure Eq. (12), we can prove the following result (the proof follows the same lines as (Lelièvre & Stoltz, 2016)[Prop. 2.3], see also (Bakry et al., 2013)[Thm 4.2.5]).

Lemma A.6. *The measure ν_Σ satisfies the Poincaré inequality Eq. (23) if and only if*

$$\|e^{t\mathcal{L}}\|_{\mathcal{B}(L_0^2(\nu_\Sigma))} \leq e^{-2\frac{L}{\beta}t}. \tag{24}$$

Exponential convergence to equilibrium is then directly obtained from Lemma A.6:

$$\begin{aligned} \|e^{t\mathcal{L}}\bar{\phi}\|_{L_0^2(\nu_\Sigma)} &\leq \|e^{t\mathcal{L}}\|_{\mathcal{B}(L_0^2(\nu_\Sigma))} \|\bar{\phi}\|_{L_0^2(\nu_\Sigma)} \\ &\leq e^{-2\frac{L}{\beta}t} \|\bar{\phi}\|_{L_0^2(\nu_\Sigma)}. \end{aligned} \tag{25}$$

This inequality implies Eq. (5) (note that $e^{t\mathcal{L}}\langle\phi\rangle_{\nu_\Sigma} = \langle\phi\rangle_{\nu_\Sigma}$) and thus proves the first assertion of Corollary A.2.

A consequence of the exponential convergence to equilibrium Eq. (25) is the following central limit theorem for time averages $\langle\phi\rangle_T = \frac{1}{T}\int_0^T \phi(q_t)\,\mathrm{d}t$ (see also (Kipnis & Varadhan, 1986)).

Theorem A.7. *(Bhattacharya, 1982) If Eq. (25) holds, then the following convergence in law is satisfied*

$$\sqrt{T}\left(\langle\phi\rangle_T - \langle\phi\rangle_{\nu_\Sigma}\right) \to \mathcal{N}(0, \sigma_\phi^2) \quad \text{as } T \to \infty,$$

where the asymptotic variance σ_ϕ^2 is given by the formula $\sigma_\phi^2 = 2\langle\bar{\phi}, -\mathcal{L}^{-1}\bar{\phi}\rangle$ with $\bar{\phi} = \phi - \langle\phi\rangle_{\nu_\Sigma}$.

To quantify the asymptotic variance, we use the following classical result.

Lemma A.8. *(e.g., (Lelièvre & Stoltz, 2016)[Prop. 2.1]) If Eq. (24) holds, then the generator $\mathcal{L}$ is invertible and the resolvent can be expressed as $-\mathcal{L}^{-1} = \int_0^\infty e^{t\mathcal{L}}\,\mathrm{d}t$ and satisfies the bound $\|\mathcal{L}^{-1}\|_{\mathcal{B}(L_0^2(\nu_\Sigma))} \leq \frac{\beta}{2L}$.*

Using Lemma A.8 and Cauchy–Schwartz inequality, the asymptotic variance in Theorem A.7 can thus be bounded as

$$\begin{aligned} \sigma_\phi^2 = 2\int_\Sigma \bar{\phi}(-\mathcal{L}^{-1}\bar{\phi})\,\mathrm{d}\nu_\Sigma &\leq 2\|\mathcal{L}^{-1}\|_{\mathcal{B}(L_0^2(\nu_\Sigma))}\|\bar{\phi}\|^2_{L_0^2(\nu_\Sigma)} \\ &\leq \frac{\beta}{L}\|\bar{\phi}\|^2_{L_0^2(\nu_\Sigma)}. \end{aligned}$$

This estimate completes the proof of the second assertion of Corollary A.2.

B. Discretization of constrained Langevin dynamics

We present here the details of the constrained training methods considered in this paper. Both the overdamped Eq. (M-7) and underdamped Eq. (M-12) Langevin dynamics are discretized for the constraints presented in Section 3. We emphasize that the initialization of each given method must be done with care: the constrained parameters, the potential slack variable, as well as their momenta in the underdamped case, have to satisfy the constraint initially.

Recall the notation introduced in Section 3: $\theta \in \mathbb{R}^n$ is the vector of all the parameters of the model, we consider the variable $q = (\theta, \xi) \in \mathbb{R}^d$, $d = n + n^\xi$, where $\xi \in \mathbb{R}^s$ is a slack variable to enforce the potential inequality constraints. The loss is extended $q = (\theta, \xi)$ as $V(q) = L_X(\theta)$ (in particular $\nabla_\xi V = 0$) and constraints are given by a map $g : \mathbb{R}^d \to \mathbb{R}^m$. The parameters are partitioned as $\theta = (\theta^u, \theta^c)$, where $\theta^u \in \mathbb{R}^{n^u}$ are not involved in any constraint while $\theta^c \in \mathbb{R}^{n^c}$ are.

B.1. Discretization of constrained overdamped Langevin (general constraint)

Following (Lelièvre et al., 2010)[Chap. 3] a simple discretization of the constrained overdamped Langevin dynamics Eq. (M-7) is given by the iteration $q_n \in \Sigma \mapsto q_{n+1}$ defined as

$$\begin{aligned} \bar{q}_{n+1} &= q_n - \nabla_q V(q_n)h + \sqrt{2\beta^{-1}h}\, R_n, \\ q_{n+1} &= \bar{q}_{n+1} - \nabla_q g(q_n)\lambda_n, \\ &\text{where } \lambda_n \in \mathbb{R}^m \text{ is such that } g(q_{n+1}) = 0, \end{aligned} \tag{26}$$

where $R_n \sim N(0, I)$ is a vector of iid standard normal random variable. The first step of Eq. (26), $\bar{q}_{n+1}$, is an Euler–Maruyama step for standard overdamped Langevin. As $\bar{q}_{n+1}$ in $\mathbb{R}^d$ is generally not on the constrained manifold Σ, the last term is present to project $\bar{q}_{n+1}$ back onto Σ, ensuring $g(q_{n+1}) = 0$. In particular, for the unconstrained parameter we have $\nabla_{\theta^u}^T g = 0_{m\times n^u}$ which implies that $\theta^u_{n+1} = \bar{\theta}^u_{n+1}$ is a standard EM step.

In general, projecting back onto the manifold Σ, i.e., finding λ_n, can be done using root-finding algorithms. Nevertheless, for certain constraints g the roots can be found explicitly. This is the case for the circle constraint Eq. (M-2) (see Section B.3). A potential weakness of method Eq. (26) is that the projection process can be guaranteed only for small enough step size h (i.e. $\bar{q}_n$ must be close to Σ). Indeed, even for the circle constraint if h is too large it might not be possible to project $\bar{q}_{n+1}$ back onto the circle following the direction $\nabla_q g(q_n)$. See (Lelièvre et al., 2020) for some discussion of methods to allow computation to be performed in the large timestep regime.

An alternative method is given by the iteration $q_n \in \Sigma \mapsto q_{n+1} \in \Sigma$ defined as in (Lelièvre et al., 2010)[Chap. 3]

$$\begin{aligned} \bar{q}_{n+1} &= q_n - \nabla_q V(q_n)\mathrm{d}t + \sqrt{2\beta^{-1}h}\, R_n, \\ q_{n+1} &= \bar{q}_{n+1} - \nabla_q g(q_{n+1})\lambda_n, \\ &\text{where } \lambda_n \in \mathbb{R}^m \text{ is such that } g(q_{n+1}) = 0, \end{aligned} \tag{27}$$

where $R_n \sim N(0, I)$ is a vector of iid standard normal random variable. The projection used in method Eq. (27) is in general more robust. The circle constraint is a good illustration of this: while in Eq. (26) we project following an oblique direction, in Eq. (27) the projection is orthogonal and always exists (see Section B.3).

B.2. Discretization of constrained underdamped Langevin (general constraint)

We next consider the discretization of the constrained underdamped Langevin dynamics Eq. (M-12) where we denote by $p = (p^u, p^c, p^\xi) \in \mathbb{R}^{n^u+n^c+n^\xi}$ the momenta associated with the configuration $q = (\theta^u, \theta^c, \xi)$. Following (Leimkuhler & Matthews, 2016), the system is split into A,B,O components Eq. (M-14), where B represents a projected impulse defined by the loss gradient (restricted to the cotangent space), O represents a projected stochastic impulse, and A represents evolution along geodesics (i.e., for circle constraints, these are rotations on the circles).

As in the overdamped case, the equality $\nabla_{\theta^u}^T g = 0_{m\times n^u}$ ensures that the unconstrained parameters and their momenta (θ^u, p^u) evolve following the A,B,O steps for unconstrained underdamped Langevin (see (Leimkuhler et al., 2016)). As the B and O components only involve a variation in the momentum p_t and because the constraint only involves q_t, they can be solved exactly for any constraint. The A component involves a variation of the configuration q_t and thus cannot be solved exactly (in law) for any constraint. However, as this part does not include any force evaluation (which would require back-propagation to compute the gradient), it can be approximated cheaply using a few steps of standard well-known schemes such as SHAKE or RATTLE (see Section B.6 for orthogonal constraints). Furthermore, for simple constraints such as the circle constraint Eq. (M-2) the A component can be solved explicitly (see Section B.4).

Let us present the details of the B and O steps. For convenience, let us introduce the following notation for the variables involved in the constraint $w = (\theta^c, \xi) \in \mathbb{R}^{n^c+n^\xi}$ and associated momentum $p^w = (p^c, p^\xi) \in \mathbb{R}^{n^c+n^\xi}$. The projection onto the cotangent space Eq. (8) is then as

$$\begin{aligned} \Pi(q) &= I_d - \begin{pmatrix} 0 & 0 \\ 0 & \Pi_w(q) \end{pmatrix}, \\ \text{with } \Pi_w &= \begin{pmatrix} g_{\theta^c}^T H^{-1} g_{\theta^c} & g_\xi^T H^{-1} g_{\theta^c} \\ g_{\theta^c}^T H^{-1} g_\xi & g_\xi^T H^{-1} g_\xi \end{pmatrix}, \end{aligned} \tag{28}$$

where we have denoted the partial Jacobians by $g_{\theta^c} = \nabla_{\theta^c}^T g \in \mathbb{R}^{m\times n^c}$, $g_\xi = \nabla_\xi^T g \in \mathbb{R}^{m\times n^\xi}$ and the matrix $H = g_{\theta^c} g_{\theta^c}^T + g_\xi g_\xi^T \in \mathbb{R}^{m\times m}$.

B component. Given $q_0, p_0 \in T^*\Sigma$ and a time $t > 0$

$$q_t = q_0, \qquad p_t = p_0 - t\nabla_q V(q_0) - \nabla_q g(q_0)(\mu_t - \mu_0),$$

where μ_t is such that $p_t \in T^*_{q_t}\Sigma$ (i.e., it satisfies the constraint $0 = \nabla_q g(q_t)p_t$). Note that as q_0, p_0 satisfy the constraints we have $\mu_0 = 0$. Projecting onto the cotangent space $T^*_{q_t}\Sigma = T^*_{q_0}\Sigma$ and using $\Pi(q_0)\nabla_q g(q_0) = 0$ and $p_0 = \Pi(q_0)p_0$, we obtain

$$\begin{aligned} p_t = \Pi(q_t)p_t &= \Pi(q_0)\big(p_0 - t\nabla_q V(q_0) - \nabla_q g(q_0)\mu_t\big) \\ &= p_0 - t\Pi(q_0)\nabla_q V(q_0). \end{aligned}$$

The B step is thus obtained for a chosen stepsize $h > 0$ as: given $q_n = (\theta^u_n, \theta^c_n, \xi_n) \in \Sigma$ and

$p_n = (p_n^u, p_n^c, p_n^\xi) \in T^*_{q_n}\Sigma$

$$
\text{(B, gen.)}\quad
\begin{aligned}
&\theta^u_{n+1} = \theta^u_n, \qquad \theta^c_{n+1} = \theta^c_n, \qquad \xi_{n+1} = \xi_n,\\
&p^u_{n+1} = p^u_n - h\nabla_{\theta^u} L_X(\theta_n),\\
&\bar{p}^c_{n+1} = p^c_n - h\nabla_{\theta^c} L_X(\theta_n), \qquad \bar{p}^\xi_{n+1} = p^\xi_n,\\
&\begin{pmatrix} p^c_{n+1} \\ p^\xi_{n+1} \end{pmatrix} = \Pi_w(w_n) \begin{pmatrix} \bar{p}^c_{n+1} \\ \bar{p}^\xi_{n+1} \end{pmatrix},\\
&\text{where } w_n = \begin{pmatrix} \theta^c_n \\ \xi_n \end{pmatrix}
\end{aligned}
\tag{29}
$$

O component. Similarly as for the B part, the O part can be solved exactly in law for any constraint. Given $q_0, p_0 \in T^*\Sigma$ and a time $t > 0$, we have

$$
\begin{aligned}
q_t &= q_0,\\
p_t &= p_0 - \gamma \int_0^t p_t \,\mathrm{d}t + \sqrt{2\gamma\tau} \int_0^t \mathrm{d}\mathcal{W}_t - \nabla_q g(q_0)\nu_t,
\end{aligned}
$$

where ν_t ensures that $p_t \in T^*_{q_t}\Sigma$. Projecting to the cotangent space $T^*_{q_t}\Sigma = T^*_{q_0}\Sigma$ as before, we obtain

$$
\begin{aligned}
p_t &= \Pi(q_t)p_t\\
&= p_0 - \gamma \int_0^t \Pi(q_0)p_t \,\mathrm{d}t + \sqrt{2\gamma\tau}\Pi(q_0) \int_0^t \mathrm{d}\mathcal{W}_t.
\end{aligned}
$$

We thus recognize that p_t is an Ornstein–Uhlenbeck process:

$$
p_t \overset{\text{law}}{=} \Pi(q_0)\big(e^{-\gamma t}p_0 + \sqrt{\tau(1-e^{-2\gamma t})}R\big)
$$

with $R \sim N(0, I_d)$, where the equality holds in law.

The O step is thus obtained for a chosen stepsize $h > 0$ as: given $q_n = (\theta^u_n, \theta^c_n, \xi_n) \in \Sigma$ and $p_n = (p^u_n, p^c_n, p^\xi_n) \in T^*_{q_n}\Sigma$

$$
\text{(O, gen.)}\quad
\begin{aligned}
&\theta^u_{n+1} = \theta^u_n, \qquad \theta^c_{n+1} = \theta^c_n, \qquad \xi_{n+1} = \xi_n,\\
&p^u_{n+1} = e^{-\gamma h}p^u_n + \sqrt{\tau(1-e^{-2\gamma h})}R^u,\\
&\bar{p}^c_{n+1} = e^{-\gamma h}p^c_n + \sqrt{\tau(1-e^{-2\gamma h})}R^c,\\
&\bar{p}^\xi_{n+1} = e^{-\gamma h}p^c_n + \sqrt{\tau(1-e^{-2\gamma h})}R^\xi,\\
&\begin{pmatrix} p^c_{n+1} \\ p^\xi_{n+1} \end{pmatrix} = \Pi_w(w_n) \begin{pmatrix} \bar{p}^c_{n+1} \\ \bar{p}^\xi_{n+1} \end{pmatrix}\\
&\text{where } w_n = \begin{pmatrix} \theta^c_n \\ \xi_n \end{pmatrix},
\end{aligned}
\tag{30}
$$

and R^u, R^c, and R^ξ are independent standard normal random variables.

B.3. Circle constraint, overdamped Langevin (c-CoLA-od)

We consider here the circle constraint Eq. (M-2), for which the partial Jacobians are computed as

$$
\begin{aligned}
\nabla_q^T g &= \big(\nabla^T_{\theta^u} g, \nabla^T_{\theta^c} g, \nabla^T_\xi g\big) \in \mathbb{R}^{m\times(n^u+n^c+m)},\\
\partial_{\theta^u_j} g_i &= 0, \;\; \partial_{\theta^c_j} g_i = 2\theta^c_i \delta_{ij}, \;\; \partial_{\xi_j} g_i = 2\xi_i\delta_{ij},
\end{aligned}
\tag{31}
$$

where δ_{ij} is the Kronecker delta.

For this constraint, the projection step in Eq. (26) can be computed explicitly. Indeed λ_n can be found by solving the m quadratic equations $0 = g_i(\bar{q}_{n+1} - \nabla_q g(q_n)\lambda_n)\; 1 \le i \le m$. The (potential) two roots of each equation corresponds to the (potential) two projections of $\bar{q}_{n+1}$ onto the circle following the direction $\nabla g_i(q_n) = 2(\theta^c_{n,i}, \xi_{n,i})$. When two roots are found, we may select the one closest to the point of origin $(\theta^c_{n,i}, \xi_{n,i})$. However, if the point to project $(\bar{\theta}^c_{n+1,i}, \bar{\xi}_{n+1,i})$ is too far away from the circle, this oblique projection may not be possible (i.e., the quadratic equation has no real root).

For the circle constraint, method Eq. (27) thus leads to a more robust projection process. Indeed, as $\nabla g_i(q_{n+1}) = 2(\theta^c_{n+1,i}, \xi_{n+1,i})$, the direction of the projection is now orthogonal to the circle. To find an expression for the orthogonal projection P of a point $(\bar{\theta}_1, \bar{\xi}_1)$ on the circle, it is easier to use a geometrical approach than to find the Lagrange multipliers:

$$
(\theta_1, \xi_1) = P(\bar{\theta}_1, \bar{\xi}_1) = \big(r_i\cos(\alpha), r_i\sin(\alpha)\big),
$$

where $\alpha = \arctan\left(\frac{\bar{\xi}_1}{\bar{\theta}_1}\right)$. We obtain the following discretization of the overdamped Langevin with circle constraints. We initialize the parameters of the neural network using standard PyTorch initialization (Paszke et al., 2017; He et al., 2015), i.e., $\mathcal{U}(-1/\sqrt{N_{in}}, 1/\sqrt{N_{in}})$, where N_{in} is the number of inputs to a layer. The auxiliary variables ξ_i corresponding to the constrained parameters θ^c_i are initialized to obey the constraint $(\theta^c_i)^2 + \xi_i^2 = r_i^2$. For a chosen stepsize $h > 0$ and given a configuration $q_n = (\theta^u_n, \theta^c_n, \xi_n) \in \Sigma$, one step of the method is defined by $q_{n+1} = (\theta^u_{n+1}, \theta^c_{n+1}, \xi_{n+1}) \in \Sigma$ as

$$
\begin{aligned}
\theta^u_{n+1,i} &= \theta^u_{n,i} - h\partial_{\theta^u_i} L_X(\theta_n) + \sqrt{2\beta^{-1}h}R^u_i,\\
\bar{\theta}^c_{n+1,i} &= \theta^c_{n,i} - h\partial_{\theta^c_i} L_X(\theta_n) + \sqrt{2\beta^{-1}h}R^c_i,\\
\bar{\xi}_{n+1,i} &= \xi_{n,i} + \sqrt{2\beta^{-1}h}R^\xi_i,\\
\alpha_{n,i} &= \arctan\left(\frac{\bar{\xi}_{n+1,i}}{\bar{\theta}^c_{n+1,i}}\right),\\
\theta^c_{n+1,i} &= r_i\cos(\alpha_{n,i}),\\
\xi_{n+1,i} &= r_i\sin(\alpha_{n,i}),
\end{aligned}
\tag{32}
$$

where R^u_i, R^c_i, R^ξ_i are independent standard normal random variables.

B.4. Circle constraint, underdamped Langevin (c-CoLA-ud)

We provide here the full discretization of the underdamped Langevin dynamics in the case of the circle constraint Eq. (M-2).

A component. For the circle constraint we can solve the A step explicitly. First recall that as $\nabla_{\theta^u}^T g = 0$, the unconstrained parameters θ^u are obtained with a standard A step of the unconstrained underdamped Langevin. Let us then focus on solving the constrained components: we denote $w = (\theta^c, \xi), p^w = (p^c, p^\xi)$. Then for $1 \leq i \leq m$ the A step in Eq. (M-14) corresponds to the constrained ODEs

$$\begin{aligned} &\dot{w}_i = p_i^w \\ &\dot{p}_i^w = -2\lambda_i w_i \\ &|\theta_i^c|^2 + |\xi_i|^2 = r_i^2, \qquad \theta_i^c p_i^c + \xi_i p_i^\xi = 0. \end{aligned} \tag{33}$$

As these constrained ODEs are uncoupled, let us drop the specification of the index i. By assumption, we are given initial conditions that satisfy the constraint $(w_0, p_0^w) \in T^*\Sigma$. Solving the second order ODE $\ddot{w} = -2\lambda w$, we find that any solution has the form $w_t = R_t^{2\lambda} w_0$, where R_t^ω is a rotation matrix with angular speed ω given with its time derivative as

$$\begin{aligned} R_t^\omega &= \begin{pmatrix} \cos(\omega t) & \sin(\omega t) \\ -\sin(\omega t) & \cos(\omega t) \end{pmatrix}, \\ \dot{R}_t^\omega &= \omega \begin{pmatrix} -\sin(\omega t) & \cos(\omega t) \\ -\cos(\omega t) & -\sin(\omega t) \end{pmatrix}. \end{aligned}$$

Computing the momentum $p_t^w = \dot{w}_t = \dot{R}_t^\omega w_0$, and using the properties of R_t^ω we verify that w_t, p_t^w satisfy the constraints in Eq. (33) ($\|.\|$ denotes the Euclidean norm in $\mathbb{R}^2$ and $\cdot$ the dot product):

$$\begin{aligned} &\|w_t\|^2 = \|R_t^\omega w_0\|^2 = \|w_0\|^2 = r^2, \\ &w_t \cdot p_t^w = w_0^T (R_t^\omega)^T \dot{R}_t^\omega w_0 = 0. \end{aligned}$$

We still have to find the angular speed $\omega = 2\lambda$ such that the momentum p_t^w is consistent with its initial value p_0^w (we denote $w_0 = (\theta_0^c, \xi_0)$ and $p_0^w = (p_0^c, p_0^\xi)$):

$$p_0^w = \dot{R}_0^\omega w_0 \quad \Leftrightarrow \quad p_0^c = \omega \xi_0 \text{ and } p_0^\xi = -\omega \theta_0^c.$$

We thus find that

$$\begin{aligned} &\xi_0 p_0^c - \theta_0^c p_0^\xi = \omega\big(|\xi_0|^2 + |\theta_0^c|^2\big) = \omega r^2 \\ \Leftrightarrow \quad &\omega = \frac{1}{r^2}\big(\xi_0 p_0^c - \theta_0^c p_0^\xi\big). \end{aligned}$$

We have thus found an explicit expression for the solution of the A component for circle constraints Eq. (33).

To complete the B and O steps given in Eq. (29) and Eq. (30), we need an explicit expression for the projection Π_w in Eq. (28) (using Eq. (31), recall that $m = n^c = n^\xi$):

$$\Pi_w(w) = \begin{pmatrix} I_m - D^{11} & -D^{12} \\ -D^{12} & I_m - D_{22} \end{pmatrix},$$

where $D^{kl} \in \mathbb{R}^{m \times m}$ are the diagonal matrices defined as

$$\begin{aligned} D_{ii}^{11} &= \frac{|\theta_i^c|^2}{|\theta_i^c|^2 + |\xi_i|^2}, \quad D_{ii}^{12} = \frac{\theta_i^c \xi_i}{|\theta_i^c|^2 + |\xi_i|^2}, \\ D_{ii}^{22} &= \frac{|\xi_i|^2}{|\theta_i^c|^2 + |\xi_i|^2}. \end{aligned}$$

Assuming that $w = (\theta^c, \xi)$ satisfies the constraint, the projection of $(\bar{p}^c, \bar{p}^\xi)$ is thus computed as

$$\begin{aligned} \begin{pmatrix} p^c \\ p^\xi \end{pmatrix} &= \Pi_w(w) \begin{pmatrix} \bar{p}^c \\ \bar{p}^\xi \end{pmatrix}, \\ \text{where} \quad p_i^c &= \bar{p}_i^c - \frac{\theta_i^c}{r_i^2}\big(\theta_i^c \bar{p}_i^c + \xi_i \bar{p}_i^\xi\big) \quad 1 \leq i \leq m, \\ p_i^\xi &= \bar{p}_i^\xi - \frac{\xi_i}{r_i^2}\big(\theta_i^c \bar{p}_i^c + \xi_i \bar{p}_i^\xi\big) \quad 1 \leq i \leq m. \end{aligned}$$

Note that in the B step Eq. (29), the above expressions can be simplified by combining the simple definition of $(\bar{p}_n^c, \bar{p}_n^\xi)$ with the constraint

$$0 = \big(\nabla^T g(q) p\big)_i = 2\big(\theta_i^c p_i^c + \xi_i p_i^\xi\big).$$

We provide below the explicit updates for the A, B and O components for circle constraints. We initialize the parameters of the net using standard PyTorch initialization (Paszke et al., 2017; He et al., 2015). The auxiliary variables ξ corresponding to the constrained parameters θ^c are initialized to obey the constraint $(\theta^c)^2 + \xi^2 = r^2$, so that $q_0 = (\theta_0^u, \theta_0^c, \xi_0) \in \Sigma$. The momenta, p^u, p^c, and p^ξ, are generated in the same manner as for standard SGD with momentum in PyTorch, i.e., as equal to the initial gradients. Subsequently, the momenta belonging to the constrained variables p^c and to the auxiliary variables p^ξ are projected using Π_w, so that $p_0 = (p_0^u, p_0^c, p_0^\xi) \in T_{q_0}^*\Sigma$. For a stepsize $h > 0$ we obtain

$$\text{(A)} \begin{cases} \theta_{n+1,i}^u = \theta_{n,i}^u + h p_{n,i}^u, \\ \omega_i = \dfrac{1}{r_i^2}\big(\xi_{n,i} p_{n,i}^c - \theta_{n,i}^c p_{n,i}^\xi\big), \\ \theta_{n+1,i}^c = \cos(\omega_i h)\theta_{n,i}^c + \sin(\omega_i h)\xi_{n,i}, \\ \xi_{n+1,i} = -\sin(\omega_i h)\theta_{n,i}^c + \cos(\omega_i h)\xi_{n,i}, \\ p_{n+1,i}^u = p_{n,i}^u, \\ p_{n+1,i}^c = \omega_i\big(-\sin(\omega_i h)\theta_{n,i}^c + \cos(\omega_i h)\xi_{n,i}\big), \\ p_{n+1,i}^\xi = -\omega_i\big(\cos(\omega_i h)\theta_{n,i}^c + \sin(\omega_i h)\xi_{n,i}\big), \end{cases}$$

$$\text{(B)}\quad \begin{cases} \theta^u_{n+1} = \theta^u_n, \qquad \theta^c_{n+1} = \theta^c_n, \qquad \xi_{n+1} = \xi_n, \\ p^u_{n+1} = p^u_n - h\nabla_{\theta^u} L_X(\theta_n), \\ \bar{p}^c_{n+1,i} = p^c_{n,i} - h\Big(1 - \dfrac{1}{r_i^2}|\theta^c_{n,i}|^2\Big)\partial_{\theta^c_i} L_X(\theta_n), \\ \bar{p}^\xi_{n+1,i} = p^\xi_{n,i} + h\dfrac{1}{r_i^2}\theta^c_{n,i}\xi_{n,i}\partial_{\theta^c_i} L_X(\theta_n), \end{cases}$$

$$\text{(O)}\quad \begin{cases} \theta^u_{n+1} = \theta^u_n, \qquad \theta^c_{n+1} = \theta^c_n, \qquad \xi_{n+1} = \xi_n, \\ p^u_{n+1} = e^{-\gamma h} p^u_n + \sqrt{\beta^{-1}(1 - e^{-2\gamma h})} R^u, \\ \bar{p}^c_{n+1} = e^{-\gamma h} p^c_n + \sqrt{\beta^{-1}(1 - e^{-2\gamma h})} R^c, \\ \bar{p}^\xi_{n+1} = e^{-\gamma h} p^c_n + \sqrt{\beta^{-1}(1 - e^{-2\gamma h})} R^\xi, \\ p^c_{n+1,i} = \Big(1 - \dfrac{|\theta^c_{n,i}|^2}{r_i^2}\Big)\bar{p}^c_{n+1,i} - \dfrac{1}{r_i^2}\theta^c_{n,i}\xi_{n,i}\bar{p}^\xi_{n+1,i}, \\ p^\xi_{n+1,i} = -\dfrac{\theta^c_{n,i}\xi_{n,i}\bar{p}^c_{n+1,i}}{r_i^2} + \Big(1 - \dfrac{|\xi_{n,i}|^2}{r_i^2}\Big)\bar{p}^\xi_{n+1,i}, \end{cases}$$

where R^u, R^c, and R^ξ are vectors of independent standard normal random variables.

B.5. Orthogonality constraint, overdamped Langevin dynamics (o-CoLA-od)

We present here a particular discretization of the constrained overdamped Langevin dynamics Eq. (M-7) for the orthogonality constraint Eq. (M-4).

For notational convenience, we present the updates for the weight matrix W^ℓ of a given layer ℓ. The updates for the biases are standard Euler–Maruyama steps such as given for θ^u in Eq. (32).

Referring to Eq. (M-4), we denote

$$\begin{array}{llll} Q = W^\ell, & r = n^\ell, & s = n^{\ell-1} & \text{if } n^{\ell-1} \le n^\ell, \\ Q = (W^\ell)^T, & r = n^{\ell-1}, & s = n^\ell & \text{otherwise.} \end{array} \tag{34}$$

so that $Q \in \mathbb{R}^{r\times s}$. With this notation, the constraint Eq. (M-4) is $g(Q) = 0$ where

$$g : \mathbb{R}^{r\times s} \to \mathbb{R}^{s\times s}, \qquad g(Q) = Q^T Q - I_s. \tag{35}$$

Recall that due to symmetry, the matrix equality $g(Q) = 0_s$ corresponds to $s(s+1)/2$ constraints. We compute the partial derivative

$$\begin{aligned} &\partial_{Q_{kl}} g_{ij}(Q) = \delta_{li} Q_{kj} + \delta_{lj} Q_{ki} \\ &1 \le i, j, k \le s,\ 1 \le l \le r. \end{aligned} \tag{36}$$

In particular, if Λ is an $s \times s$ symmetric matrix, we verify that

$$\sum_{i,j=1}^{s} \partial_{Q_{kl}} g_{ij}(Q)\Lambda_{ij} = 2\big(Q\Lambda\big)_{kl}.$$

We thus obtain the natural matrix form of the constrained dynamics Eq. (M-7): $Q_t : (0,\infty) \to \mathbb{R}^{r\times s}$ solves

$$\begin{aligned} &\mathrm{d}Q_t = -\nabla_Q V(Q_t)\,\mathrm{d}t + \sqrt{2\beta^{-1}}\,\mathrm{d}\mathcal{W}_t - Q_t\,\mathrm{d}\Lambda_t, \\ &g(Q_t) = 0, \end{aligned} \tag{37}$$

where $\big(\nabla_Q V\big)_{ij} = \partial_{Q_{ij}} V = \partial_{W^\ell_{ij}} L_X$ (or $\partial_{W^\ell_{ji}} L_X$) and $\mathcal{W}_t$ is a Wiener process in $\mathbb{R}^{r\times s}$. Furthermore the process Λ_t has values in the $s \times s$ symmetric matrices and is the Lagrange multiplier corresponding to the $s(s+1)/2$ constraints.

Applying discretization scheme Eq. (26) to Eq. (37), we obtain the iteration step $Q_n \in \Sigma \mapsto Q_{n+1} \in \Sigma$ given by

$$\begin{aligned} &\bar{Q}_{n+1} = Q_n - h\nabla_Q V(Q) + \sqrt{2\beta^{-1}h} R_n, \\ &Q_{n+1} = \bar{Q}_{n+1} - Q_n\Lambda_n, \end{aligned} \tag{38}$$

where Λ_n is a symmetric $s \times s$ matrix s.t. $g(Q_{n+1}) = 0$ and $R_n \in \mathbb{R}^{r\times s}$ is a matrix of independent standard normal random variables.

Note that the projection step in Eq. (38) requires to solve a non-linear system. Following a similar technique as described in (Leimkuhler & Reich, 2004)[Chap. 8], we derive a quasi-Newton scheme for that task. Using the fact that Q_n satisfies the constraint we verify that

$$\bar{Q}^T_{n+1} Q_n = I_s - h\nabla_Q V(Q_n)^T Q_n + \sqrt{2\beta^{-1}h} R_n^T Q_n.$$

The constraint $g(Q_{n+1}) = 0$ thus reads

$$\begin{aligned} 0 &= \big(\bar{Q}_{n+1} - Q_n\Lambda_n\big)^T\big(\bar{Q}_{n+1} - Q_n\Lambda_n\big) - I_s \\ &= \big(\bar{Q}^T_{n+1}\bar{Q}_{n+1} - I_s\big) - 2\Lambda_n + \mathcal{O}(\sqrt{h}), \end{aligned} \tag{39}$$

where $\mathcal{O}(\sqrt{h})$ denotes a matrix whose 2-norm has order $\sqrt{h}$. Solving for Λ_n, we find

$$\Lambda_n = \frac{1}{2}\big(\bar{Q}^T_{n+1}\bar{Q}_{n+1} - I_s\big) + \mathcal{O}(\sqrt{h}).$$

Neglecting the terms of order $\sqrt{h}$ and higher, we obtain the following quasi-Newton scheme: setting $Q^{(0)} = \bar{Q}_{n+1}$, repeat the iteration

$$\begin{aligned} &Q^{(k+1)} = Q^{(k)} - Q_n\Lambda^{(k)}, \\ &\text{where } \Lambda^{(k)} = \frac{1}{2}\big((Q^{(k)})^T Q^{(k)} - I_s\big), \end{aligned} \tag{40}$$

until the process reaches convergence and set $Q_{n+1} = Q^{(k+1)}$. To assess whether convergence has been reached, a tolerance on the 2-norm of $\Lambda^{(k)}$ can be assigned: $\|\Lambda^{(k)}\| \le \mathrm{TOL}$. However in practice, to ensure that the process ends and to avoid undesirable overhead we typically prefer to either combine this stopping criterion with a limit for the number K of iterations, or use a fixed number of iterations

K. Note that estimate Eq. (39) ensures that a small number of iterations K is sufficient for the constraint to be satisfied up to a small error.

The initialization for the constrained weights is performed following (Saxe et al., 2013), which is an built-in option in PyTorch. Other parameters are initialized using the standard PyTorch initialization (Paszke et al., 2017; He et al., 2015) unless otherwise indicated. Constraints are applied layer-wise, where for convolutional layers with weight tensors of the size $n_l \times n_{l-1} \times n_h \times n_w$ (where n_h and n_w are the height and width of the kernel) the weight matrices are reshaped as $n_l \times n_{l-1} n_h n_w$. For CNNs these reshaped matrices are typically rectangular. If they are thin, but long (i.e., $n_l > n_{l-1} n_h n_w$) we apply the constraint $W^T W = I$, but if they have more columns than rows we apply the constraint $W W^T = I$.

B.6. Orthogonality constraint, underdamped Langevin (o-CoLA-ud)

To discretize the underdamped Langevin constrained dynamics, we need the orthogonal projection Π onto the cotangent space $T^*_Q\Sigma$. As the constraint Eq. (35) is given in a matrix form, using the formula Eq. (8) is not very convenient so we will rather derive Π from its projection property.

Using Eq. (36), we find that for $1 \leq i, j \leq s$

$$0 = \sum_{k=1}^{s} \sum_{l=1}^{r} \partial_{Q_{kl}} g_{ij}(Q) P_{kl} = (P^T Q + Q^T P)_{ij},$$

which leads to the following convenient expression for the cotangent space

$$T^*_Q\Sigma = \{P \in \mathbb{R}^{r\times s} \mid P^T Q + Q^T P = 0_s\}.$$

Now, given $\bar{P} \in \mathbb{R}^{r\times s}$ we want to find a symmetric $s \times s$ matrix Λ such that $P = \bar{P} - Q\Lambda$ belongs to $T^*_Q\Sigma$, i.e.,

$$0_s = P^T Q - Q^T P = \bar{P}^T Q + Q^T \bar{P} - \Lambda Q^T Q - Q^T Q \Lambda.$$

This equation is easily solved for $Q \in \Sigma$ and we find $\Lambda = \frac{1}{2}(\bar{P}^T Q + Q^T \bar{P})$. We obtain the following expression for the projection onto the cotangent space:

$$\Pi_Q : \mathbb{R}^{r\times s} \to \mathbb{R}^{r\times s},$$
$$\bar{P} \mapsto \Pi_Q \bar{P} = \bar{P} - \frac{1}{2} Q(\bar{P}^T Q + Q^T \bar{P}).$$

We then verify that Π_Q is indeed a projection onto the cotangent space $T^*_Q\Sigma$ (i.e., $\Pi_Q \bar{P} \in T^*_Q\Sigma \ \forall \bar{P} \in \mathbb{R}^{r\times s}$ and $\Pi_Q^2 = \Pi_Q$) and that this projection is orthogonal with respect to the Frobenius inner product on $\mathbb{R}^{r\times s}$ (i.e., $\langle \bar{P} - \Pi_Q \bar{P}, P\rangle = 0$, where $\langle A, B\rangle = \mathrm{tr}(A^T B)$).

A component. For the orthogonal constraint, the A component in Eq. (M-14) can only be solved approximately. A simple yet efficient discretization of A is the RATTLE scheme (see e.g. (Leimkuhler & Reich, 2004)[Chap. 8]):

$$\begin{aligned}
&Q_{n+1} = Q_n + hP_{n+1/2},\\
&P_{n+1/2} = P_n - Q_n\Lambda_{n+1/2}\\
&\quad \text{where } \Lambda_{n+1/2} \text{ is s.t. } Q_{n+1}^T Q_{n+1} = I_s,\\
&P_{n+1} = P_{n+1/2} - Q_{n+1}\Lambda_{n+1}\\
&\quad \text{where } \Lambda_{n+1} \text{ is s.t. } Q_{n+1}^T P_{n+1} + P_{n+1}^T Q_{n+1} = 0_s.
\end{aligned} \tag{41}$$

Denoting $\bar{\Lambda}_{n+1/2} = h\Lambda_{n+1/2}$, $\bar{P}_{n+1} = P_{n+1/2}$ and using the projection operator Π_Q, Eq. (41) can be rewritten as

$$\begin{aligned}
&\bar{Q}_{n+1} = Q_n + hP_n,\\
&Q_{n+1} = \bar{Q}_{n+1} - Q_n\bar{\Lambda}_{n+1/2}\\
&\quad \text{where } \bar{\Lambda}_{n+1/2} \text{ is s.t. } Q_{n+1}^T Q_{n+1} = I_s \quad \text{(use Eq. (40))},\\
&\bar{P}_{n+1} = P_n - \frac{1}{h} Q_n\bar{\Lambda}_{n+1/2}, \qquad P_{n+1} = \Pi_{Q_{n+1}} \bar{P}_{n+1}.
\end{aligned} \tag{42}$$

As in the overdamped case, we may now use the quasi-Newton scheme Eq. (40) for the projection step (to approximate $\bar{\Lambda}_{n+1/2}$). Using K iterations of the quasi-Newton scheme Eq. (40) (i.e., $Q_{n+1} = Q^{(K)}$), we verify that $-Q_n\bar{\Lambda}_{n+1/2}$ satisfies

$$\begin{aligned}
-Q_n\bar{\Lambda}_{n+1/2} &= \sum_{k=0}^{K-1} Q_n\Lambda^{(k)} = \sum_{k=0}^{K-1} Q^{(k+1)} - Q^{(k)}\\
&= Q^{(K)} - Q^{(0)} = Q_{n+1} - \bar{Q}_{n+1},
\end{aligned}$$

so that $\bar{P}_{n+1} = P_n + \frac{1}{h}(Q_{n+1} - \bar{Q}_{n+1})$.

We obtain the following full discretization of the underdamped Langevin dynamics with orthogonality constraint. The initialization for the constrained weights is performed following (Saxe et al., 2013). Corresponding momenta are initialized as the initial gradients (equivalently to standard PyTorch initialization) and subsequently projected using $P_0 = \bar{P}_0 - \frac{1}{2} Q_0(\bar{P}_0^T Q_0 + Q_0^T \bar{P}_0)$. The A,B,O steps are then given as:

$$\text{(A, OG)} \begin{cases}
\bar{Q}_{n+1} = Q_n + hP_n, \quad Q^{(0)} = \bar{Q}_{n+1},\\
\text{for } k = 0 \text{ to } K-1:\\
\qquad Q^{(k+1)} = Q^{(k)} - Q_n\Lambda^{(k)},\\
\quad \text{where } \Lambda^{(k)} = \frac{1}{2}\Big(\big(Q^{(k)}\big)^T Q^{(k)} - I_s\Big),\\
Q_{n+1} = Q^{(K)},\\
\bar{P}_{n+1} = P_n + \frac{1}{h}\big(Q_{n+1} - \bar{Q}_{n+1}\big),\\
P_{n+1} = \Pi_{Q_{n+1}} \bar{P}_{n+1} = \bar{P}_{n+1}\\
\quad - \frac{1}{2} Q_{n+1}\Big(\bar{P}_{n+1}^T Q_{n+1} + \big(Q_{n+1}\big)^T \bar{P}_{n+1}\Big).
\end{cases}$$

$$\text{(B, OG)}\begin{cases} Q_{n+1} = Q_n, \\ \bar{P}_{n+1} = P_n - h\nabla_Q V(Q_n), \\ P_{n+1} = \Pi_{Q_n} P_{n+1} \\ = \bar{P}_{n+1} - \frac{1}{2} Q_n \Big(\bar{P}_{n+1}^T Q_n + \big(Q_n\big)^T \bar{P}_{n+1} \Big), \end{cases}$$

$$\text{(O, OG)}\begin{cases} Q_{n+1} = Q_n, \\ \bar{P}_{n+1} = e^{-\gamma h} P_n + \sqrt{\beta^{-1}(1 - e^{-2\gamma h})} R_n, \\ P_{n+1} = \Pi_{Q_n} \bar{P}_{n+1} \\ = \bar{P}_{n+1} - \frac{1}{2} Q_n \Big(\bar{P}_{n+1}^T Q_n + \big(Q_n\big)^T \bar{P}_{n+1} \Big), \end{cases}$$

where R_n is a matrix of independent standard normal random variables.

C. Feedforward neural network notations and gradients (backpropagation)

Given a dataset $X = \{x_i, y_i\}_{i=1}^N$, where $x_i \in \mathbb{R}^{d^{\text{in}}}, y_i \in \mathbb{R}^{d^{\text{out}}}$, we want to construct an interpolant of the relation $x_i \mapsto y_i$. For this task, we choose a feedforward neural network (NN) with $L+1$ layers (i.e., L parametrized layers, L is the *depth*). For $1 \le \ell \le L$ we denote the *width* of layer ℓ as d^ℓ ($d^0 = d^{\text{in}}$, $d^L = d^{\text{out}}$). The parameters of the NN at layer ℓ are given by the weights and biases

$$W^\ell \in \mathbb{R}^{d^\ell \times d^{\ell-1}}, \quad b^\ell \in \mathbb{R}^{d^\ell} \qquad 1 \le \ell \le L.$$

For notational convenience, let us stack the parameters in a vector

$$\theta^\ell = \begin{pmatrix} \theta_W^\ell \\ \theta_b^\ell \end{pmatrix}, \quad \theta_b^\ell = b^\ell \in \mathbb{R}^{d^\ell},$$

$$\theta_W^\ell = \text{vect}(W^\ell) = \begin{pmatrix} W^\ell e_1 \\ \vdots \\ W^\ell e_{d^{\ell-1}} \end{pmatrix} \in \mathbb{R}^{d^\ell d^{\ell-1}}.$$

In particular $\theta^\ell \in \mathbb{R}^{n^\ell}$, where n^ℓ is the number of parameters in layer ℓ, $n^\ell = d^\ell \times d^{\ell-1} + d^\ell$. The vector of all parameters is denoted $\theta = (\theta^1, \dots, \theta^\ell) \in \mathbb{R}^n$, where $n = \sum_{l=1}^L n_\ell$.

Each layer $1 \le \ell \le L$ is equipped with an *activation function* $\varphi^\ell : \mathbb{R}^{d^\ell} \to \mathbb{R}^{d^\ell}$, which is is applied component wise: $\varphi_i^\ell(x) = \phi^\ell(x_i)$, for some $\phi^\ell : \mathbb{R} \to \mathbb{R}$. In each layer $1 \le \ell \le L$, we define the following functions

$$a^\ell : \mathbb{R}^{n^\ell \times d^{\ell-1}} \to \mathbb{R}^{d^\ell}, \quad a^\ell(\theta^\ell, z^{\ell-1}) = W^\ell z^{\ell-1} + b^\ell,$$
$$z^\ell : \mathbb{R}^{n^\ell \times d^{\ell-1}} \to \mathbb{R}^{d^\ell}, \quad z^\ell(\theta^\ell, z^{\ell-1}) = \varphi^\ell(a^\ell(\theta^\ell, z^{\ell-1})),$$

to which we associate the following shorthand notation

$$a_{\theta^\ell}^\ell = a^\ell(\theta^\ell, \cdot) : \mathbb{R}^{d^{\ell-1}} \to \mathbb{R}^{d^\ell},$$
$$z_{\theta^\ell}^\ell = z^\ell(\theta^\ell, \cdot) : \mathbb{R}^{d^{\ell-1}} \to \mathbb{R}^{d^\ell}.$$

We verify that the map $\theta^\ell \mapsto a^\ell(\theta^\ell, z^{\ell-1})$ can be written as

$$\begin{aligned} a^\ell(\theta^\ell, z^{\ell-1}) &= \big((z^{\ell-1})^T \otimes I_{d^\ell}\big)\theta_W^\ell + \theta_b^\ell \\ &= \big((z^{\ell-1})^T \otimes I_{d^\ell}, I_{d^\ell}\big)\theta^\ell, \end{aligned}$$

where I_d denotes the identity matrix in $\mathbb{R}^d$ and for $z \in \mathbb{R}^s$, $z^T \otimes I_d = (z_1 I_d, \dots, z_s I_d)$. We then introduce the *intermediate classifiers* as $p^0(x) = x$ and

$$p^\ell : \mathbb{R}^{n \times d^0} \to \mathbb{R}^{d^\ell}, \quad p^\ell(\theta, x) = z_{\theta^\ell}^\ell \circ \cdots \circ z_{\theta^1}^1(x),$$

$1 \le \ell \le L$, for which we use the shorthand $p_\theta^\ell = p^\ell(\theta, \cdot)$. The (final) classifier is then the function $p_\theta = p_\theta^L : \mathbb{R}^{d^{\text{in}}} \to \mathbb{R}^{d^{\text{out}}}$.

To train the NN on the dataset X, we define the *loss function* as

$$L_X : \mathbb{R}^n \to \mathbb{R} \qquad L_X(\theta) = -\sum_{i=1}^N D\big(p(\theta, x_i), y_i\big),$$

where $D = D(\hat{y}, y) : \mathbb{R}^{d^{\text{out}}} \times \mathbb{R}^{d^{\text{out}}} \to \mathbb{R}$ is a function that measures the discrepancy between $\hat{y}$ and y. In a simple classification case, $d^{\text{out}} = 1$ and D is chosen to be the cross-entropy. All the commonly used training method require the computation of the gradient of the loss function given as

$$\nabla_\theta L_X : \mathbb{R}^n \to \mathbb{R}^n$$
$$\nabla_\theta L_X(\theta) = -\sum_{i=1}^N \nabla_{\hat{y}} D\big(p(\theta, x_i), y_i\big) \nabla_\theta p(\theta, x_i).$$

Expression for the gradient of the loss (backpropagation)

Recall that we denote the Jacobian matrix of a function $f : \mathbb{R}^n \to \mathbb{R}^m$ as the map $\nabla^T f : \mathbb{R}^n \to \mathbb{R}^{m \times n}$ defined as $(\nabla^T f)_{ij} = \partial_j f_i$. Given two functions $f : \mathbb{R}^{m^1} \to \mathbb{R}^{m^2}$ and $g : \mathbb{R}^{m^2} \to \mathbb{R}^{m^3}$ the chain rule implies the Jacobian matrix of the composition $g \circ f$ satisfies

$$\nabla^T(g \circ f) : \mathbb{R}^{m^1} \to \mathbb{R}^{m^3 \times m^1},$$
$$x \mapsto \nabla^T(g \circ f)(x) = \nabla^T g\big(f(x)\big) \nabla^T f(x).$$

We compute the partial Jacobians of $a^j(\theta^j, z^{j-1})$ as

$$\nabla_{\theta^j}^T a^j : \mathbb{R}^{n^j} \times \mathbb{R}^{d^{j-1}} \to \mathbb{R}^{d^j \times n^j},$$
$$\nabla_{\theta^j}^T a^j(\theta^j, z^{j-1}) = \big((z^{j-1})^T \otimes I_{d^j}, I_{d^j}\big), \tag{43}$$

and

$$\nabla_{z^{j-1}}^T a^j : \mathbb{R}^{n^j} \times \mathbb{R}^{d^{j-1}} \to \mathbb{R}^{d^j \times d^{j-1}}$$
$$\nabla_{z^{j-1}}^T a^j(\theta^j, z^{j-1}) = W^j. \tag{44}$$

The partial Jacobians of $z^j(\theta^j, z^{j-1})$ are then

$$\begin{aligned}
&\nabla^T_{\theta^j} z^j : \mathbb{R}^{n^j} \times \mathbb{R}^{d^{j-1}} \to \mathbb{R}^{d^j \times n^j},\\
&\nabla^T_{\theta^j} z^j(\theta^j, z^{j-1}) = \nabla^T_{a^j}\varphi^j\big(a^j_{\theta^j}(z^{j-1})\big)\nabla^T_{\theta^j} a^j(\theta^j, z^{j-1}),\\
&\nabla^T_{z^{j-1}} z^j : \mathbb{R}^{n^j} \times \mathbb{R}^{d^{j-1}} \to \mathbb{R}^{d^j \times d^{j-1}}, \nabla^T_{z^{j-1}} z^j(\theta^j, z^{j-1})\\
&= \nabla^T_{a^j}\varphi^j\big(a^j_{\theta^j}(z^{j-1})\big)\nabla^T_{z^{j-1}} a^j(\theta^j, z^{j-1}),
\end{aligned} \tag{45}$$

where we note that $\big(\nabla_{a^j}\varphi^j(z)\big)_{rs} = \partial_t\phi^j(z_r)\delta_{rs}$ (i.e., the matrix is diagonal).

The partial Jacobians of the classifier are then given by

$$\begin{aligned}
\nabla^T_{\theta^\ell} p(\theta, x) &= \nabla^T_{z^{L-1}} z\big(\theta^L, p^{L-1}_\theta(x)\big)\cdots\\
&\quad \cdots \nabla^T_{z^\ell} z^{\ell+1}\big(\theta^{\ell+1}, p^\ell_\theta(x)\big)\nabla^T_{\theta^\ell} z^\ell\big(\theta^\ell, p^{\ell-1}_\theta(x)\big),\\
&\quad 1 \le \ell \le L-1,\\
\nabla^T_{\theta^L} p(\theta, x) &= \nabla^T_{\theta^L} z^L\big(\theta^L, p^{L-1}_\theta(x)\big),
\end{aligned} \tag{46}$$

and

$$\begin{aligned}
\nabla^T_x p(\theta, x) &= \nabla^T_{z^{L-1}} z^L\big(\theta^L, p^{L-1}_\theta(x)\big)\\
&\quad \cdots \nabla^T_{z^1} z^2\big(\theta^2, p^1_\theta(x)\big)\nabla^T_x z^1\big(\theta^1, x\big).
\end{aligned} \tag{47}$$

From (47), replacing the partial Jacobians of z^j with the expressions provided in (45), we obtain

$$\nabla^T_x p(\theta, x) = F^L_x W^L \cdots F^2_x W^2 F^1_x W^1, \tag{48}$$

where F^j_x is the Jacobian matrix of the activation in the jth layer, φ^j (e.g., if $\varphi^j = \mathrm{ReLU}$, F^j_x is a diagonal matrix with 1 and 0 entries). Constraining the weights moreover has a direct influence on the smoothness of the interpolant $p_\theta(x)$.

From (46), replacing the partial Jacobians of z^j with the expressions provided in (45), we obtain

$$\begin{aligned}
\nabla^T_{\theta^L} p_\theta(x) &= F^L_x P^L_x,\\
\nabla^T_{\theta^\ell} p_\theta(x) &= F^L_x W^L \cdots F^{\ell+1}_x W^{\ell+1} F^\ell_x P^\ell_x,\\
&\quad 1 \le \ell \le L-1,
\end{aligned} \tag{49}$$

where matrices F^j_x are defined above and P^j_x is sparse with repeated entries of $p^j_\theta(x) = z^j_{\theta^j} \circ \cdots \circ z^1_{\theta^1}(x)$. This shows that as the depth L is increased, the gradient of $p_\theta(x)$ with respect to the parameters of any layer is composed of sparse products of the weights W^j. This multiplicative structure leads to difficulty of DNN training: the multiplication of small weights $\ll 1$ leads to a low value of the gradient which in turn has the effect of slowing the training (*vanishing gradient*), while the multiplication of large weights $\gg 1$ leads to a large value of the gradient which affects the stability of the learning procedure (*exploding gradient*).

Let us explain the stability in more detail. As training methods are discretization of a dynamics involving the gradient $\nabla_\theta L_X$, the stability of a method is connected to the Lipschitz constant L on the statespace $E = \mathbb{R}^n$ of the gradient.[1] Assuming that L_X is twice differentiable, the largest L can be is

$$M \le \sup_{\theta \in E} \big|\lambda_{\max}(\theta)\big|, \tag{50}$$

where $\lambda_{\max}(\theta)$ denotes the largest eigenvalue of the Hessian $\nabla^2_\theta L_X(\theta)$. The entries of the Hessian are computed as

$$\begin{aligned}
&\big(\nabla^2_\theta L_X(\theta)\big)_{rs} =\\
&\quad \sum_{i=1}^N \Big(\nabla_\theta p(\theta, x_i)\nabla^2_{\hat{y}} D\big(p(\theta, x_i), y_i\big)\nabla^T_\theta p(\theta, x_i)\Big)_{rs}\\
&\quad + \sum_{k=1}^{d^{\mathrm{out}}} \partial_{\hat{y}_k} D\big(p(\theta, x_i), y_i\big)\partial^2_{\theta_r\theta_s} p_k(\theta, x_i).
\end{aligned}$$

Even without providing the heavy expression of $\partial^2_{\theta_r\theta_s} p_k$, using (46) in this expression allow to appreciate the impact of the magnitudes of the weights and of the depth on the Hessian and thus on the stability.

D. Additional Numerical Details and Results

We perform all experiments using PyTorch (Paszke et al., 2017) on NVIDIA DGX-1 GPUs. We compare our constrained methods with PyTorch's SGD with momentum optimiser. Unless otherwise indicated, we use for SGD $h = 0.1$ and $mom = 0$ (to compare with our constrained overdamped Langevin method) or $mom = 0.9$ (to compare with our constrained underdamped Langevin method). We use standard PyTorch initialization for all unconstrained parameters (He et al., 2015; Paszke et al., 2017). Below we provide implementation details for all our experiments.

D.1. Orthogonality Constraints

A plot of the planar spiral data set binary classification problem as used to produce Figure 4 and Figure 5 is provided in Figure D1. The first class of the data set is generated using

$$\begin{aligned}
x &= 2\sqrt{t}\cos(8\sqrt{t}\pi) + 0.02\mathcal{N}(0,1),\\
y &= 2\sqrt{t}\sin(8\sqrt{t}\pi) + 0.02\mathcal{N}(0,1),
\end{aligned} \tag{51}$$

where t is drawn repeatedly from the uniform distribution $\mathcal{U}(0,1)$ to generate data points. The other class of this dataset is obtained by shifting the argument of the trigonometric functions by π. For our experiments we used 500 training data, 1000 test data points and 5% subsampling.

To generate the results presented in Figure 4 and 5 of the main paper, which show the effect of orthogonality con-

[1] Recall that the Lipschitz constant of a function $h : E \subset R^r \to \mathbb{R}^s$ is the smallest constant M such that $|f(x) - f(y)| \le M|x - y|$ for all $x, y \in E$, where $|.|$ denotes the Euclidean norm.

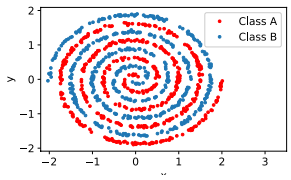

Figure D1. Data set generated using Eq. (51).

straints on this spiral data set (see Fig. D1), we use multi-layer perceptrons with ReLU activation and binary cross entropy (BCE) loss. In our experiments we vary the number of 100-node hidden layers of the multi-layer perceptrons. To compare the performance of our o-CoLA-od constrained method with standard SGD we set the temperature $\tau = 0$ and $h = 0.1$ for all methods to generate Fig. 4. For Fig. 5 we do a grid-search to find the optimal value of the penalty strength for the orthogonal regularization approach with respect to the stepsize. In Fig. D2 we show the effect of using a small temperature perturbation τ = 1e-6. The size of the temperature parameter was chosen to approximately match observed fluctuations in the loss function. A more precise parameterization is left for a subsequent work.

We also applied our orthogonality-constrained methods to the ResNet-34 architecture on CIFAR-10 image classification data (Krizhevsky & Hinton, 2009), see Figure 6. The input data is pre-processed using random crop (pad=4), random horizontal flip, and normalization. In this setting, running SGD with orthogonal initialization worsened the generalization performance of the resulting net and hence the standard PyTorch initialization was used for SGD. We train for 150 epochs and use a batchsize of 128. In Figure D3 we compare the overdamped variant o-CoLA-od (with $\tau = 0$) to its unconstrained counterpart. We observe that the use of an orthogonality constraint gives lower test loss throughout training.

D.2. Circle constraints

For the results shown in Figure 1, Figure 2, Figure 3, and Table 1 the first class of the data set is generated using

$$x = \sqrt{t}\cos(4\sqrt{t}\pi) + 0.05\mathcal{N}(0,1),$$
$$y = \sqrt{t}\sin(4\sqrt{t}\pi) + 0.05\mathcal{N}(0,1), \qquad (52)$$

where t is repeatedly drawn from $\mathcal{U}(0,1)$. The other class is obtained by shifting the argument of the trigonometric functions by π. For our experiments we used 100 training data points, 2000 test data points and 2% subsampling. We use a 500-node single hidden layer perceptron, with ReLU activation and BCE loss. We choose the optimal weight decay value for SGD through line search. The results in Fig. 3 were obtained by computing the gradient of the predictions of a trained classifier (after 10,000 epochs) on a 1000x1000 grid using second order accurate central differences.

For our Fashion-MNIST (Xiao et al., 2017) example we reduce the number of training data samples to 10,000 and we increase the number of test data samples to 60,000. We use a 1000-node SHLP with ReLU activation, cross entropy loss and batchsize 128. Our main result with our circle constrained approach is presented in Figure 7, the accompanying mean test accuracies with standard deviations are: 87.63±0.04% (c-CoLA-ud), 87.39 ±0.06% (SGD), 87.47 ±0.38% (SGD with WD = 1e-4), 87.29 ±0.58% (SGD with WD = 5e-5), 87.45 ±0.06% (SGD with WD = 1e-5). Hyperparameters SGD: $h = 0.1, mom = 0.8$. Hyperparameters c-CoLA-ud: $h = 0.3, \gamma = 1, r_0 = 0.05, r_1 = 0.1, \tau = 0$.

In Table D2 we present extensive hyperparameter tests for the test accuracy and test loss obtained after 400 epochs (averaged over 5 runs) using SGD-m with and without weight decay (WD). In Figure D4 we show that both the test loss and the maximum magnitude of the weights of the network remains small and stable throughout training for our circle constrained approach, while SGD shows signs of overfitting.

We also evaluate the performance of a small transformer model (Vaswani et al., 2017) on the Penn Treebank (Marcus et al., 1993) and Wikitext-2 (Merity et al., 2017) data. The transformer has 2 encoder layers. Each encoder layer consists of self-attention with 2 heads and a feedforward network with 200 nodes followed by layer norms. We use batchsize 1024 for the Penn Treebank data and batchsize 128 for the Wikitext-2 dataset. We present the lowest validation loss obtained in 200 epochs by SGD-m and our circle constrained method c-CoLA-ud in Table 2 of the main paper. In Table D1 we provide a comparison with weight decay.

Table D1. Minimum validation loss on Penn Treebank and Wikitext-2 using a transformer trained using SGD-m. We found weight decay set to WD = 1e-4 to give the best results for SGD-m. In comparison, the transformer trained using c-CoLA-ud obtains a minimum validation loss of 4.81 (Penn Treebank) and 5.09 (Wikitext-2). Using c-CoLA-ud therefore outperforms standard SGD-m in the case without WD, but does less well than SGD-m with weight decay, if the magnitude of the weight decay has been carefully tuned. In contrast, for Fashion-MNIST image data using a MLP we find that c-CoLA-ud outperforms both SGD-m with weight decay and SGD-m without weight decay (see Table D2).

Optimizer	Penn Treebank	Wikitext-2
Without WD		
mom = 0.7	4.87	5.13
mom = 0.8	4.83	5.13
mom = 0.9	4.84	5.13
With WD		
mom = 0.7	4.84	5.01
mom = 0.8	4.77	5.02
mom = 0.9	4.77	5.02

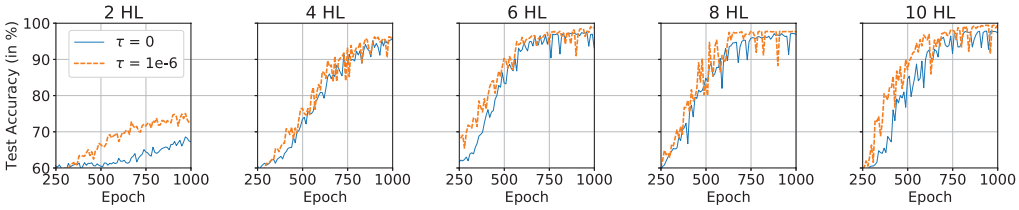

Figure D2. The effect of temperature on the performance of the o-CoLA-od optimizer for the 4-turn spiral data set (same set-up as for Fig. 4). MLPs with varying numbers of hidden layers (HL) were trained using o-CoLA-od with $h = 0.1$ and either $\tau = 0$ (blue line) or $\tau =$ 1e-6 (orange line). Results are averaged over 5 runs. The use of temperature is shown to speed up training and often slightly increases the obtained test accuracies.

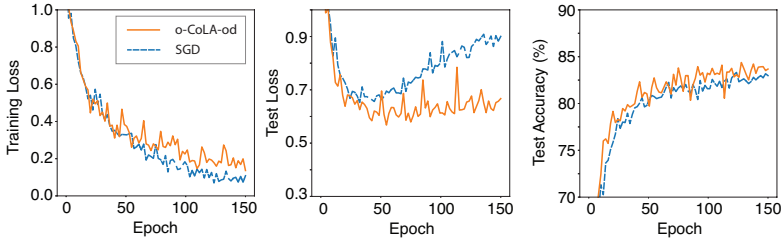

Figure D3. Training loss (left), test loss (middle) and test accuracy (right) of a ResNet-34 trained using SGD vs. o-CoLA-od on CIFAR-10 data, $h = 0.1$ (averaged over 5 runs). The orthogonality constraint provides modestly higher test accuracy and inhibits overfitting.

Table D2. These results are obtained for the Fashion-MNIST dataset using SGD with momentum to train a 1000-node SHLP. The results presented in the two right-hand columns are all obtained with weight decay set to 1e-4. We found this value to give the best results for SGD-m during a hyperparameter search. In comparison to the results for SGD-m shown in this table our circle constrained net reaches test accuracy **87.63%**, with test loss **0.386** without using weight decay (see Figure 7). Hence it outperforms standard SGD with momentum both with and without weight decay.

SGD with mom		no WD		with WD	
		Test Acc.	Test Loss	Test acc.	Test Loss
h = 0.2	mom = 0.8	87.18%	1.06	84.05%	0.696
	mom = 0.7	87.38%	0.890	87.0%	0.547
h = 0.1	mom = 0.9	86.97%	1.133	85.35%	0.634
	mom = 0.8	87.39%	0.824	87.47%	0.531
	mom = 0.7	87.39%	0.750	87.25%	0.517
h = 0.05	mom = 0.95	86.67%	1.226	85.63%	0.623
	mom = 0.9	87.33%	0.837	86.24%	0.569
	mom = 0.8	87.27%	0.719	87.33%	0.511

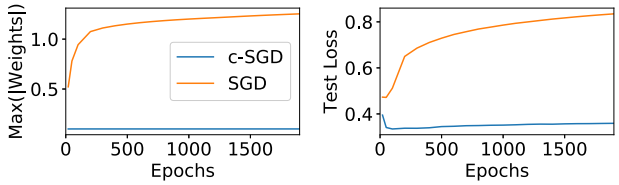

Figure D4. Result is obtained for the Fashion-MNIST dataset with the same hyperparameter settings as in Figure 7 of the main paper. We observe that the maximum absolute size of weights in the output layer of the network (left) and test loss (right) remain small and stable throughout training for the circle constrained method (c-SGD or c-CoLA-ud). In contrast SGD shows clear signs of overfitting.

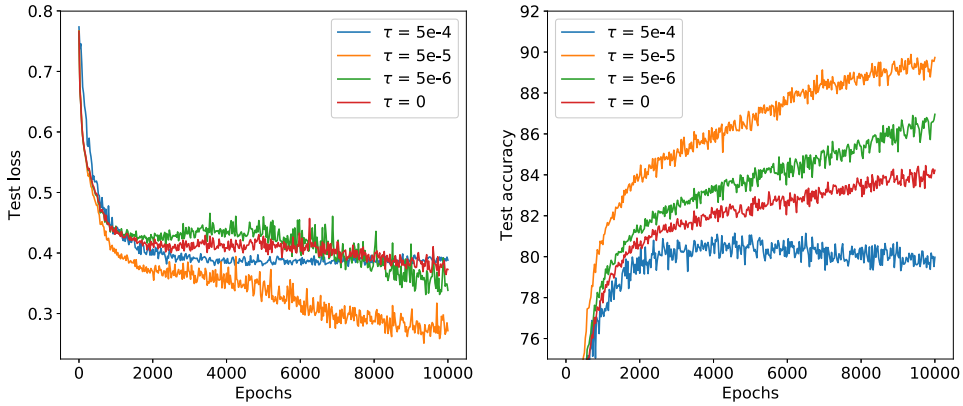

Figure D5. Test loss and test accuracy averaged over 100 runs for constrained approaches with varying levels of additive noise, i.e., with different values of the temperature hyperparameter τ. The set-up is the same as for Fig. 1 and Fig. 2 in the main paper, i.e., we train a 500-node single hidden layer perceptron for a spiral binary classification problem (Eq. (52)). Hyperpar. settings: $h = 0.05$, 2% subsampling, $r_0 = 1, r_1 = 5$ (see Eq. (M-2)). The best performance is obtained using temperature τ = 5e-5. This is also the temperature that results in the classifier with the lowest curvature estimate (see Table D3).

D.3. Curvature

It is difficult to establish a commonly agreed definition of curvature for a boundary that is potentially non-differentiable at a finite number of points. We computed our curvature estimates using the method described below which we suggest is indicative of the curvature of the locally smoothed classification boundary and allows us to compare the relative curvature estimates of classifiers trained using different optimizers.

We evaluate the smoothness properties of our trained classifiers after a fixed number of 10,000 epochs. The curvature of a level curve $\phi(x, y) = 0$ is defined as (Persson, 2006):

$$\kappa = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} = \frac{\phi_{xx}\phi_y^2 - 2\phi_y\phi_x\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}}$$

However, since we do not have access to the exact form of ϕ, we fit a contour to the model's predictions on a 1000x1000 grid using matplotlib.pyplot.contour, which returns an array containing the coordinates of points along the contour. We view these as discrete samples from the parametric curve $(x(t), y(t))$. The gradients of these are computed using second order accurate central differences. This can then be used to compute the approximate curvature (Gray et al., 2006):

$$\kappa = \frac{|x''y' - x'y''|}{(x'^2 + y'^2)^{3/2}} \tag{53}$$

Although this results in a rough estimate, by averaging our results over 100 runs, we suggest this gives us some insight on relative curvature estimates of classifiers trained using different optimizers.

In Table D3 and Figure D5 we study the effect of varying the temperature hyperparameter τ, which controls the additive noise level (see Eq. (M-7)), on the generalization performance and curvature of the resulting classifiers on the spiral dataset defined by Eq. (52). We show that there appears to be an ideal choice of temperature (in this case τ = 5e-5), for which the best generalization performance is obtained using our circle constrained approach. We also show that the trained classifier which has the lowest curvature estimate also obtains the best generalization performance.

Table D3. Same set-up as for Fig. 1, 2, and Figure D5. We present the mean curvature, standard deviation (std), and maximum (max) curvature of classifier boundaries obtained using our constrained approach with different values of the temperature τ. The lowest curvature is obtained using τ = 5e-5, which also corresponds to the classifier which obtains the best generalization performance (see Figure D5). These results are averaged over 100 runs.

	Curvature Approximation		
τ for C-SGLD	Mean	Std	Max
τ = 0	9.38	317	$5.58 \cdot 10^5$
τ = 1e-6	9.01	273	$1.63 \cdot 10^6$
τ = 5e-6	7.75	166	$5.86 \cdot 10^5$
τ = 1e-5	7.06	108	$4.06 \cdot 10^5$
τ = 5e-5	6.08	40.8	$1.43 \cdot 10^5$
τ = 1e-4	7.62	178	$9.47 \cdot 10^5$
τ = 5e-4	15.9	850	$5.07 \cdot 10^6$