A1 Proofs and Derivations

A1.1 Proof of Lemma 1

Step 1 First, we derive representation (10). We start from the proposed variational drift (9) with identity rescaling, i.e.

$$a^Z(x,t) = a^X(x) + v(t)T(x)\,. \tag{A1}$$

with $v : [0, t_f] \to \mathbb{R}^{n\times m}$ and $T : \mathbb{R}^n \to R^m$. If we denote the columns of v by $v_1, \dots, v_m$, the control part of Eq. (A1) may be written as

$$v(t)T(x) = \sum_{i=1}^{m} v_i(t)T_i(x) \tag{A2}$$

where T_i represents the components of the vector T. The drift (A1) induces a family of Ito processes parametrized by the deterministic, time-dependent function v given by the SDE

$$\mathrm{d}Z_t = \left(a^X(Z_t) + v(t)T(Z_t)\right)\mathrm{d}t + b(Z_t)\mathrm{d}W_t\,. \tag{A3}$$

Inserting the control (A1) into the objective function (8), the prior drift a^X cancels and the divergence between variational process and prior becomes

$$D_{\mathrm{KL}}[P^Z \,||\, P^X] = \frac{1}{2}\int_0^{t_f} \mathsf{E}\left[T(Z_t,t)^\top v(t)^\top D(Z_t,t)^{-1} v(t)T(Z_t,t)\right]\mathrm{d}t \tag{A4}$$

To proceed, let us rewrite the tensor contraction within the expectation above using the expanded form of the control (A2). For clarity, we also suppress the arguments and get

$$T^\top v^\top D^{-1} vT = \sum_{i,j} v_i T_i D^{-1} T_j v_j \tag{A5}$$

Now let us define a block matrix function $\psi : \mathbb{R}^n \to \mathbb{R}^{nm\times nm}$ and a vector $u \in \mathbb{R}^{nm}$ as

$$\psi(x) = \begin{pmatrix} T_1T_1D^{-1} & \dots & T_1T_mD^{-1} \\ \vdots & \dots & \vdots \\ T_mT_1D^{-1} & \dots, & T_mT_mD^{-1} \end{pmatrix}, \quad u = \begin{pmatrix} v_1 \\ \vdots \\ v_m \end{pmatrix}.$$

With this, the tensor contraction (A5) can be written as

$$T^\top v^\top D^{-1} vT = u^\top \psi u\,. \tag{A6}$$

Now observe that only the elements of ψ depend on the stochastic process Z_t. Consequently, we arrive at the following form of the divergence

$$D_{\mathrm{KL}}[P^Z \,||\, P^X] = \frac{1}{2}\int_0^{t_f} u(t)^\top \mathsf{E}[\psi(Z_t)]u(t)\mathrm{d}t\,. \tag{A7}$$

This function contains expectations of the form $\mathsf{E}[T_i(Z_t)T_j(Z_t)D^{-1}_{kl}(Z_t))]$. By augmenting the summary statics S with these quantities, we can find a function g such that $\psi(t) = g(\varphi(t))$. While this choice is always possible, there may often be more convenient ways to construct g. Using g, we can write

$$D_{\mathrm{KL}}[P^Z \,||\, P^X] = \frac{1}{2}\int_0^{t_f} u(t)^\top g(\varphi(t))u(t)\mathrm{d}t\,. \tag{A8}$$

as given in (10).

Step 2 What is left to show is that (A7) is a quadratic form in u. For this, we have to show that the matrix $\psi(t)$ is symmetric positive semi-definite for almost every $t \in [0, t_f]$. The function ψ is symmetric by construction. Consequently, $E[\psi(Z_t)]$ is symmetric. Now fix $u \in \mathbb{R}^{nm}$ and let v be the corresponding matrix representation. Set

$$H(v,t) = T(x,t)^\top v(t)^\top D^{-1}(x,t) v(t) T(x,t)\,.$$

Then $H(v,t) \geq 0$ for almost all $x \in \mathbb{R}^n$ and $t \in [0, t_f]$. To see this, we understand $v(t)T(x,t)$ as a vector in $\mathbb{R}^n$. The result follows because $D(x,t)$ is p.s.d. almost everywhere by assumption. Here, the strict definiteness is lost in general. For example, if $m > n$ we may find v such that $v(t)T(x,t) = 0$. Now since H and A represent the same quadratic form, A has to be positive semi-definite as well.

Step 3 So far, we have not considered rescaling. Now let $R : \mathbb{R}^n \to R^{n \times n}$ be an invertible matrix valued function and consider the rescaled controlled drift

$$a^Z(x,t) = a^X(x) + R(x)v(t)T(x,t)\,.$$

Evaluating the prior contribution to the divergence yields

$$D_{\mathrm{KL}}[P^Z \,||\, P^X] = \frac{1}{2}\int_0^{t_f} \mathsf{E}\left[T(Z_t)^\top v(t)^\top \underbrace{R(Z_t)^\top D(Z_t,t)^{-1} R(Z_t)}_{\equiv \tilde{D}^{-1}(Z_t)} v(t) T(Z_t)\right] \mathrm{d}t\,. \tag{A9}$$

We can now directly repeat steps 1 and 2 by replacing $D(x) \to \tilde{D}(x)$ in the construction of g.

A1.2 Proof of Proposition 1

The full variational inference problem for the proposal class parametrized by u is given by

$$\min_u \frac{1}{2}\int_0^{t_f} u(t)^\top g(\varphi(t)) u(t)\,\mathrm{d}t - \sum_{k=1}^n \mathsf{E}[\log p(y_k \mid Z_{t_k})] + \log C\,. \tag{A10}$$

Eq. (A10) is a direct consequence of (8) and Lemma 1 and may be understood as a stochastic control problem. We now assume that the expected log-likelihood may be expressed in terms of the expected summary statistics φ, i.e. $\mathsf{E}[\log p(y_k \mid Z_{t_k})] = F(\varphi(t_k))$. We also ignore the evidence $\log C$ which is independent of u. This leads to the streamlined representation

$$\min_u \frac{1}{2}\int_0^{t_f} u(t)^\top g(\varphi(t)) u(t)\,\mathrm{d}t - \sum_{k=1}^n F(\varphi(t_k))\,. \tag{A11}$$

The objective function in (A11) corresponds to the negative ELBO for the proposal family parametrized by u. Unfortunately, the simple appearance of (A11) is deceiving since $\varphi(t) = \mathsf{E}[S(Z_t)]$ implicitly depends on u. Now recall that $\varphi(t)$ obeys an evolution equation of the form

$$\frac{d}{dt}\varphi(t) = \mathsf{E}[A^\dagger S(Z_t)]\,.$$

We can now convert (A11) into a constrained problem

$$\begin{aligned} \min_{u,\varphi} \quad & \frac{1}{2} t_f u(t)^\top g(\varphi(t)) u(t)\,\mathrm{d}t - \sum_{k=1}^n F_i(\varphi(t_k)) \\ \text{s.t.} \quad & \dot{\varphi}(t) = \mathsf{E}[A^\dagger S(Z_t)] \end{aligned}\,. \tag{A12}$$

We emphasize that (A12) is an equivalent representation of the variational problem and does not contain any approximations beyond the choice of the variational family. This also means that it has the full complexity of the original stochastic control problem (A10). In order to obtain a more tractable problem, we use a moment closure on the constraint to get an ODE of the form

$$\frac{d}{dt}\varphi(t) = f(u, \varphi(t))\,. \tag{A13}$$

With this moment closure relaxation, the variational problem (A12) reduces to the deterministic control problem given in the main text.

Comment In general, when a moment closure is employed, there is no global Markov process corresponding exactly to the closed moment equations. Then, the objective function is not a true lower bound of the evidence but an approximate lower bound. Such behavior is well-known for other structured variational approximation, e.g. for cluster variational methods Yedidia et al. (2000); Wainwright and Jordan (2008). Since moment closure introduces an additional approximation, results of the moment-based variational inference have to be checked empirically. However, we do not see this as a major problem since such empirical verification is required for all forms of variational inference anyway.

A1.3 Proof of Lemma 2

Consider to stochastic processes Z and Z' over an interval $[0, t_f]$ that are members of the variational family as defined by the drift and Eq. (9). Let Z and Z' be parametrized by u and u', respectively. Inserting Z and Z' into the general path divergence of diffusion processes (7) , we get

$$D_{\mathrm{KL}}[Z \,||\, Z'] = \frac{1}{2}\int_0^{t_f} \mathsf{E}\left[T(Z_t)^\top \left(v'(t)-v(t)\right)^\top D^{-1}(Z_t,t)\left(v'(t)-v(t)\right)T(Z_t)\right]dt$$

This expression is the same as (A4) with v replaced by $v - v'$. Therefore, we may follow along the same lines as in Sec. A1.1 and obtain

$$D_{\mathrm{KL}}[Z \,||\, Z'] = \frac{1}{2}\int_0^{t_f} \left(u'(t)-u(t)\right)^\top g(\varphi(t))\left(u'(t)-u\right)dt\,.$$

To get the suggested representation, set $G : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ as

$$G(u)[u'-u, u'-u] = \int_0^{t_f} \left(u'(t)-u(t)\right)^\top g(\varphi(t))\left(u'(t)-u\right)dt\,.$$

What is left is to verify the properties of G. Bilinearity follows immediately from the above definition. The symmetry and positive definiteness follow from Lemma 1.

A1.4 Proof of Proposition 2

Consider the variational problem of the main text given by

$$u^* = \arg\min_u J[u]\,. \tag{A14}$$

Here, we understand J as a functional of the controls u only. This is possible because u together with an initial condition fully defines the moments φ. We would like to perform a steepest descent that respects the metric of the manifold on which u lives. We therefore replace (A14) by the sequential optimization problem

$$u^{(i+1)} = \arg\min_u J[u]\,, \quad \text{s.t.} \quad \frac{1}{2}G(u^{(i)})(u-u^{(i)}, u-u^{(i)}) = \epsilon\,.$$

The following calculation is inspired by the discussion of neighboring optimal solutions in Stengel (1994). For sufficiently small ϵ, we may linearize J around the current estimate $u^{(i)}$. Enforcing the constraint via a Lagrange multiplier, we obtain the unconstrained functional

$$J'[u,\lambda] = J[u^{(i)}] + \delta J[u^{(i)}, u-u^{(i)}] + \lambda\left(\frac{1}{2}G(u^{(i)})(u-u^{(i)}, u-u^{(i)}) - \epsilon\right)\,.$$

Here δJ denotes the Gateaux derivative of J. For simplicity, set $\delta u = u - u^{(i)}$. We consider δu as a small perturbation of $u^{(i)}$ and denote as $\delta\varphi$ the deviation from $\varphi^{(i)}$ induced by δu. It turns out that $\delta\varphi$ satisfies the linearized forward equation

$$\delta\dot{\varphi}(t) = f_\varphi(u^{(i)}(t), \varphi^{(i)}(t))\delta\varphi(t) + f_u(u^{(i)}(t), \varphi^{(i)}(t))\delta u\,. \tag{A15}$$

The linearized contribution δJ is given by

$$\delta J[u^{(i)}, u-u^{(i)}] = \int_0^{t_f} L_\varphi(u^{(i)}(t), \varphi^{(i)}(t))\delta\varphi(t)\mathrm{d}t + \int_0^{t_f} L_u(u^{(i)}(t), \varphi^{(i)}(t))\delta u(t)\mathrm{d}t$$

where φ is understood as a function of δu determined by (A15). We therefore obtain the objective function

$$\begin{aligned} J'[u,\lambda] = &\int_0^{t_f} L_\varphi(u^{(i)}(t),\varphi^{(i)}(t))\delta\varphi(t)\mathrm{d}t + \int_0^{t_f} L_u(u^{(i)}(t),\varphi^{(i)}(t))\delta u(t)\mathrm{d}t \\ &+ \frac{\lambda}{2}\int_0^{t_f} \delta u(t)^\top \psi(\varphi^{(i)}(t))\delta u(t)\mathrm{d}t + \text{const}\,. \end{aligned} \tag{A16}$$

that we have to minimize subject to the constraint (A15). We can now follow the standard variational procedure to obtain an adjoint equation

$$\dot\eta(t) = L_\varphi(u^{(i)}(t),\varphi^{(i)}(t)) - f_\varphi^\top(u^{(i)}(t),\varphi^{(i)}(t))\eta(t) \tag{A17}$$

that satisfies the reset conditions

$$\eta(t_k^-) = \eta(t_k^+) + \frac{d}{d\varphi}F(\varphi(t_k)), \quad k=1,\dots,n \tag{A18}$$

at the observation times. In addition, we obtain the algebraic constraint

$$0 = L_u(u^{(i)}(t),\varphi^{(i)}(t)) - f_u(u^{(i)}(t),\varphi^{(i)}(t))\eta(t) + \lambda\psi(\varphi^{(i)}(t))\delta u(t)\,. \tag{A19}$$

In contrast to the non-linearized problem, the stationarity conditions decouple in this case. This means we can solve for the controls explicitly. Denoting the solution of (A17) as $\eta^{(i)}$ the solution can be expressed as

$$\delta u^{(i)}(t) = -\frac{1}{\lambda}g^{-1}(\varphi^{(i)}(t))\left(L_u(u^{(i)}(t),\varphi^{(i)}(t)) - f_u(u^{(i)}(t),\varphi^{(i)}(t))\eta^{(i)}(t)\right) \tag{A20}$$

Now we also know that $L_u(u^{(i)}(t),\varphi^{(i)}(t)) = g(\varphi^{(i)}(t))u^{(i}(t)$. Thus, we get the natural gradient update steps as

$$u^{(i+1)}(t) = u^{(i)}(t) + \delta u^{(i)}(t) = u^{(i)}(t) - h\left(u^{(i)}(t) - \psi^{-1}(\varphi^{(i)}(t))f_u(u^{(i)}(t),\varphi^{(i)}(t))\eta^{(i)}(t)\right)\,. \tag{A21}$$

Here, we also introduced the step size $h = \frac{1}{\lambda}$ that is determined by ϵ. To recover the regular gradient, we use that gradient descent is a steepest descent with respect to the Euclidean metric. In our function space setting, the natural analogue is the L_2 norm. Therefore, gradient descent is obtained by repeating the above calculations for

$$G(u)(u'-u,u'-u) = \int_0^{t_f}(u'(t)-u(t))^\top(u'(t)-u(t))\,\mathrm{d}t\,.$$

Thus, the only thing we have to change is replacing g^{-1} in (A20) with the identity matrix.

Comments In a typical application, we will initialize the descent with all controls set to zero. This setting recovers the prior process. Intuitively, we can see the natural gradient descent as a smooth transition from the prior process to the (locally) best posterior approximation within the variational family. We note that due to moment closure, we only have access to approximate moments φ. Therefore, the natural gradient is also an approximation to the true natural gradient.

A1.5 Recovering the Gaussian Process Approximation

For this section, we assume that the diffusion term b does not depend on the state. The Gaussian process approximation only requires first and second order moments. We therefore choose

$$S(x) = (x_1,\dots,x_n,x_1^2,x_1x_2,x_2^2,\dots,x_n^2)^\top\,.$$

The GP approximation is defined by a linear time-dependent drift. In order to recover this within our framework, we need to find T such that

$$a^Z(x,t) = a(x) + v(t)T(x) \overset{!}{=} u_0(t) + u_1(t)x$$

where $u_0 : [0, t_f] \to \mathbb{R}^n$ and $u_1 : [0, t_f] \to \mathbb{R}^{n\times n}$. Now consider the choice

$$T(x,t) = \begin{pmatrix} 1 \\ x \\ a(x,t) \end{pmatrix}, \quad v(t) = \begin{pmatrix} u_0(t) & u_1(t) & u_2(t) \end{pmatrix}$$

understood as block matrix notation with $u_2 : [0, t_f] \to \mathbb{R}^{n\times n}$. This will lead to a variational drift of the form

$$a^Z(x,t) = a(x) + u_0(t) + u_1(t)x + u_2(t)a(x)\,.$$

If we fix u_2 to the constant function with output minus one, the prior drift will cancel and we have constructed a linear GP. Using the general moment equation (2) with the drift a^Z, we can now derive the moment equations for $\varphi(t) = \mathsf{E}[S(Z_t)]$. Represented in terms represented in terms of m, $\bar{M}$ we get

$$\begin{aligned} \dot{m}(t) &= u_0(t) + u_1(t)m(t)\,, \\ \dot{\bar{M}}(t) &= u_1(t)\bar{M}(t) + \bar{M}(t)u_1(t)^\top + D\,. \end{aligned} \tag{A22}$$

These equations are the standard equations for mean and variance of a linear GP. Eq. (A22) defines the forward function f required for implementation in our framework. The second function required is L or g. Here, L is a bit more convenient and is given by

$$L(u(t), \varphi(t)) = \mathsf{E}[(u_0(t) + u_1(t)x - a(Z_t))^\top D^{-1}(u_0(t) + u_1(t)Z_t - a(Z_t))]\,.$$

After a few algebraic multiplications, we observe that the only model dependent quantities required for L are $\mathsf{E}[a(Z_t]$, $\mathsf{E}[a(Z_t)Z_t^\top]$ and $\mathsf{E}[a(Z_t)a(Z_t)^\top]$ expressed as functions of m and $\bar{M}$.

A2 Additional Information

A2.1 Moment Closure Approximations

In the main part, we discussed how two obtain moment equations for Markov processes and gave a general idea on moment closure. Here, we will discuss strategies to obtain such a closure scheme, i.e. how to find the function h such that we can proceed from (4) to (5). We focus on distributional closures that correspond to a projection onto a given parametric family (Bronstein and Koeppl, 2018). A distributional closure is constructed by picking a parametric proposal distribution q_ϕ on the state space $\mathcal{X}$. To obtain a closure scheme, the first step is to find $\phi(t)$ such that

$$\mathsf{E}_{\phi(t)}[R(X)] = \varphi(t)\,, \tag{A23}$$

where $\varphi(t) = \mathsf{E}[S(Z_t)]$ are the expected summary statistics used to approximate the process. Eq. (A23) defines a valid moment closure when the conditions of the implicit function theorem are satisfied. Assuming we have obtained $\varphi(t)$, we can evaluate

$$h(\varphi(t)) \equiv \mathsf{E}_{\phi(t)}[R(X)]\,, \tag{A24}$$

where the expectation is taken with respect to $q_{\phi(t)}$.

It has also been shown that moment closure tends to work better if the support of the proposal distribution matches the support of the target process. Here, we focus on two simple probabilistic closures: the multivariate normal closure for processes defined on $\mathbb{R}^n$ and the multivariate log-normal closure for processes defined on $\mathbb{R}^n_+$. Another advantage of these two schemes is that they correspond directly to first and second order moments and may thus be used as a starting point before investigating more advanced schemes. To simplify the presentation, we denote the first order moments as $m \equiv \mathsf{E}[X]$, the second order moments as $M \equiv \mathsf{E}[XX^\top]$ and the second order central moments as $\bar{M} \equiv \mathsf{E}[(X-m)(X-m)^\top]$. We write general powers in multi-index form, i.e.

$$X^\alpha \equiv \prod_{i=1}^n X_i^{\alpha_i}\,.$$

This will be useful to represent general power moments. We also define $k \equiv \sum_{i=1}^n \alpha_i$ as the order of the α-th multi-moment.

Multivariate Normal Clouse Let $X \sim \mathcal{N}(\mu, \Sigma)$ be a multivariate normal random variable on $\mathbb{R}^n$. Since the distribution is fully specified by the man μ and the covariance Σ, all moments of the form $\mathsf{E}[X^\alpha]$ with $\alpha \in \mathbb{N}^n$ can be expressed as functions of μ and Σ. One way to obtain such a representation is via Isserlis' theorem (Isserlis, 1918). We will follow an alternative approach via the moment generating function.

Lemma A1. *Let X be as above and $\alpha \in \mathbb{R}^n$ a multi-index. Then*

$$\mathsf{E}[X^\alpha] = \prod_{i=1}^{n} \frac{\partial^{\alpha_i}}{\partial s_i^{\alpha_i}} M(s) \bigg|_{s=0}$$

where

$$M(s) = \mathsf{E}[\exp(s^\top X)] = \exp\left(\mu^\top s + \frac{1}{2} s^\top \Sigma s\right)$$

is the moment generating function of $\mathcal{N}(\mu, \Sigma)$.

While Lemma (A1) does not provide an explicit formula for direct numerical implementation, it is straightforward to automatically generate the closure relations using a computer algebra system. In particular, we automatically construct moment equations using Sympy (Meurer et al., 2017) and convert them to PyTorch functions.

Multivariate Lognormal Closure A log-normal random variable can be obtained by exponential transform of a normal random variable. This generalizes to the multivariate case. More specifically, let $Z \sim \mathcal{N}(\mu, \Sigma)$. Then we say

$$X = \exp(Z)$$

has a log-normal distribution. Here, the exponential is understood as acting component-wise.

Lemma A2. *Let X be as above and $\alpha \in \mathbb{R}^n$ a multi-index. Then*

$$\mathsf{E}[X^\alpha] = \left(\prod_i \frac{\mathsf{E}[X_i]^{2\alpha_i}}{\mathsf{E}[X_i^2]^{\frac{\alpha_i}{2}}}\right) \left(\prod_{i,j} \frac{\mathsf{E}[X_i X_j]^{\frac{\alpha_i \alpha_j}{2}}}{\mathsf{E}[X_i]^{\frac{\alpha_i \alpha_j}{2}} \mathsf{E}[X_j]^{\frac{\alpha_i \alpha_j}{2}}}\right)$$

This result can be shown by exploiting that $\mathsf{E}[X^\alpha] = \mathsf{E}\left[\exp\left(\alpha^\top Z\right)\right]$ corresponds to the moment generating function of a normal distribution. As a consequence of Lemma A2, we obtain the explicit closure function

$$\mathrm{Cl}(m, M, \alpha) = \left(\prod_i \frac{m_i^{2\alpha_i}}{M_i^{\frac{\alpha_i}{2}}}\right) \left(\prod_{i,j} \frac{M_{ij}^{\frac{\alpha_i \alpha_j}{2}}}{m_i^{\frac{\alpha_i \alpha_j}{2}} m_j^{\frac{\alpha_i \alpha_j}{2}}}\right). \tag{A25}$$

The log-normal closure (A25) can be implemented efficiently using tensor operations. It is also differentiable and thus suitable for backpropagation.

A2.2 Rescaling

Consider a drift without rescaling of the form (A1). While leading to a convenient quadratic objective function, this form of the control has two major drawbacks. The first drawback is that the matrix-valued function g in (10) is of dimension $(n \cdot m, n \cdot m)$ with n, m corresponding to the dimension of the state space and the number of control features, respectively. Assuming the number control features is proportional to the dimension, the function g requires $\mathcal{O}(n^4)$ elements. The second problem is specific to models with state-dependent diffusion term. In this case, the elements of g are of the form $\mathsf{E}[T_i(Z_t)T_j(Z_t)D_{kl}^{-1}(Z_t)]$. This expression requires an analytic inverse of the diffusion tensor which is rarely available. We now discuss two special choices of rescaling. First, consider a case where the diffusion term b is known analytically and set $R = b$. We then have

$$\tilde{D}^{-}1(x) \equiv b(x)^\top D^{-1}(x) b(x) = b(x)^\top \left(b(x) b(x)^\top\right)^{-1} b(x) = I\,.$$

The matrix g now only depends on moments of the form $\mathsf{E}[T_i(Z_t, t)T_j(Z_t, t)]$. As a consequence, the objective function becomes independent of the diffusion tensor. In addition, the number of non-zero elements of g now scales as $\mathcal{O}(n^2)$. The second choice of rescaling aims at the case where we are provided with D rather than b and thus consider $R = D$ leading to

$$\tilde{D}^{-1}(x) = D(x)^\top D^{-1}(x) D(x) = D(x,t)\,. \tag{A26}$$

While this choice does not improve the scaling, g now depends on expressions of the form $\mathsf{E}[T_i(Z_t)T_j(Z_t)D_{kl}(Z_t)]$ such that we get rid of the inverse diffusion tensor. It also has an interesting intuitive interpretation. Recall that the true posterior drift is given by $\bar{a}(x,t) = a(x,t) + D(x,t)\nabla \log(\beta(x,t))$. Thus, the ansatz corresponds to approximating the log-transformed backward function by a linear feature model.

A2.3 A Standard Approximation

Inspired by the variational Gaussian process approximation, we would like to construct a method that approximates the data-driven term by a feedback control linear in the state and requires first and second order moments. This corresponds to the choices

$$\begin{aligned} S(x) &= (x_1, \dots, x_n, x_1^2, x_1x_2, x_2^2, \dots, x_n^2)^\top\,, \\ T(x) &= (1, x_1, \dots, x_n)^\top\,. \end{aligned}$$

For this special class, it is convenient to represent the control in terms of functions u_0, u_1 such that we can write

$$v(t)T(x) = u_0(t) + u_1(t)x \tag{A27}$$

A short calculation shows that the moment equations are given by

$$\begin{aligned} \dot{m}(t) &= \mathsf{E}[a(Z_t)] + \mathsf{E}[R(Z_t)]u_0(t) + \mathsf{E}(R(Z_t)u_1(t)Z_t] \\ \dot{M}(t) &= E[a(Z_t)Z_t^\top] + E[Z_t a(Z_t)^\top] + \mathsf{E}[D(Z_t)] \\ &\quad + \mathsf{E}[R(Z_t)u_0 Z_t^\top] + \mathsf{E}[Z_t u_0 Z_t^\top R(Z_t)^\top] \\ &\quad + \mathsf{E}[R(Z_t)u_1(t)Z_t Z_t^\top] + \mathsf{E}[Z_t Z_t^\top u_1(t)^\top R(Z_t)^\top] \end{aligned}$$

A2.4 Specific Subclasses

In this section, we present moment equations for special classes and more specific models considered in the experiment section.

Constant Diffusion For models with constant diffusion terms b, we can always choose the corresponding rescaling. In combination with the approach outlined in A2.3, we obtain the moment equations

$$\begin{aligned} \dot{m}(t) &= \mathsf{E}[a(Z_t)] + bu_0(t) + bu_1(t)m(t)]\,, \\ \dot{M}(t) &= E[a(Z_t)Z_t^\top] + E[Z_t a(Z_t)^\top] + bb^\top \\ &\quad + bu_0(t)m(t)^\top + m(t)u_0(t)^\top b^\top + bu_1(t)M(t) + M(t)u_1(t)^\top b^\top\,. \end{aligned}$$

The second function required is the contribution to the KL-divergence that can be provided in terms of L or g (c.f. Thm. 1). Here, using L is more convenient and we get

$$L(u(t), m(t), M(t)) = \frac{1}{2}\left(u_0(t)^\top u_0(t) + 2u_0(t)^\top u_1(t)m(t) + \mathrm{Tr}(u_1(t)^\top u_1(t)M(t))\right)\,.$$

Since the last equation is model-independent, we have implemented a base class from which custom models can be derived. In particular, implementation of a given model only requires custom functions for $\mathsf{E}[a(Z_t)]$ and $E[a(Z_t)Z_t^\top]$. For polynomial drift functions, the required expectations can be evaluated straightforwardly using our Gaussian moment closure implementation in Sympy (c.f. Lemma A1).

Population Models We consider a general population model defined by the chemical Langevin equation (25). Here, the diffusion term $b(x)$ is not given and we have only access to the diffusion tensor $D(x) = V^\top \mathrm{diag}(h(X_t))V$. We will therefore choose a linear control with rescaling $R = D$. In the regime where the CLE is valid, we typically have $X_i \gg 1$. We will also restrict the discussion to physically plausible systems with $s_{ik} \leq 2$. Under these conditions, the propensity functions can be approximated as

$$h_i(x) = c_i \prod_{k=1}^{d} x_k^{s_{ik}} \,. \tag{A28}$$

For the proposed variational process class, we obtain the moment equations

$$\begin{aligned} m_i(t) &= V_{ji}\mathsf{E}[h_j(Z_t)] + V_{ki}V_{kj}u_{0,j}(t)\mathsf{E}[h_k(Z_t)] + V_{li}V_{lj}u_{1,jk}(t)\mathsf{E}[h_l(Z_t)Z_{t,k}] \\ M_{ij}(t) &= V_{ki}\mathsf{E}[h_k(Z_t)Z_{t,j}] + V_{kj}\mathsf{E}[h_k(Z_t)Z_{t,i}] + V_{ki}V_{kl}u_{0,l}(t)\mathsf{E}[h_k(Z_t)Z_{t,j}] + V_{kj}V_{kl}u_{0,l}(t)\mathsf{E}[h_k(Z_t)Z_{t,i}] \\ &\quad + V_{ki}V_{kl}u_{1,lm}(t)\mathsf{E}[h_k(Z_t)Z_{t,m}Z_{t,j}] + V_{kj}V_{kl}u_{1,lm}(t)\mathsf{E}[h_k(Z_t)Z_{t,m}Z_{t,i}] + V_{ki}V_{kj}\mathsf{E}[h_k(Z_t)] \end{aligned} \tag{A29}$$

While these expressions may look unwieldy, we observe that when the propensities are modeled by (A28), all expectations in (A28) are of the form $\mathsf{E}[Z_t^\alpha]$ and can be easily evaluated with the generic log-normal closure function A25. In addition, expectations required for the function g are of the same form.

Multivariate Geometric Brownian Motion For the multivariate Brownian motion model described in the main text, we consider a linear control with rescaling b. We obtain the moment equations

$$\begin{aligned} \dot{m}_i(t) &= r_i m_i(t) + m_i(t)R_{ij}u_{0,j}(t) + R_{ij}u_{1,jk}M_{ik}(t)\,, \\ \dot{M}_{ij} &= r_i M_{ij}(t) + r_j M_{ji}(t) + R_{ik}u_{0,k}M_{ij}(t) + R_{jk}u_{0,k}M_{ji}(t) \\ &\quad + R_{ik}u_{1,kl}\mathsf{E}[Z_{t,i}Z_{t,j}Z_{t,l}] + R_{jk}u_{1,kl}\mathsf{E}[Z_{t,i}Z_{t,j}Z_{t,l}] + D_{ij}M_{ij}(t) \end{aligned} \tag{A30}$$

where we have used Einstein sum convention. In order to obtain closed equations, we compute the third order moments $E[Z_{t,i}Z_{t,j}Z_{t,l}]$ via the general log-normal closure formula given in Lemma A2. Since for the rescaling with b, the second function g becomes independent of the model, (A30) is the only model specific quantity required for implementation.

A2.5 Parameter Inference

A2.5.1 Extended Control Problem

Variational parameter inference corresponds to maximizing the evidence lower bound jointly with respect to the model parameters θ and variational parameters u. If we instead minimize the negative ELBO, we obtain the optimization problem

$$\min_{\theta,u} \quad \underbrace{D_{\mathrm{KL}}[P^Z \,||\, P^X] - \sum_{k=1}^{n} \mathsf{E}[\log p(y_k \mid Z_{t_k})]}_{\equiv J[\theta,u]} \,. \tag{A31}$$

We can now do the same reductions as for the derivation of the smoothing control problem

$$\begin{aligned} \min_{\theta,u,\varphi} \quad & \frac{1}{2}\int_0^{t_f} u(t)^\top g(\theta,\varphi(t))u(t)\,\mathrm{d}t - \sum_{k=1}^{n} F_i(\varphi(t_k)) \\ \text{s.t.} \quad & \dot{\varphi}(t) = \mathsf{E}[A^\dagger S(Z_t)] \end{aligned} \,. \tag{A32}$$

Of course, the dynamic constraint depends on θ as well. Three methods are commonly used to solve the variational inference problem (A31).

Coordinate Descent Starting from an initial guess $\theta^{(0)}$, $u^{(0)}$ the updates are computed as

$$\begin{aligned} u^{(i+1)} &= \min_u J[\theta^{(i)}, u]\,, \\ \theta^{(i+1)} &= \min_\theta J[\theta, u^{(i+1)}]\,. \end{aligned}$$

This is the classical variational expectation maximization(VEM) algorithm used in mean field variational inference. It is most effective when the updates can be computed in closed form. In the scenario considered here, this will typically not be the case. Even if possible, obtaining the closed form updates requires model specific calculations that we try to avoid. We therefore do not consider the VEM any further.

Gradient Descent In Prop. 2, we have presented regular and natural gradient descent in the controls to solve the smoothing problem. The proof can be extended straightforwardly to include a gradient update with respect to θ as well. More explicitly, the parameter updates take the form

$$\theta^{(i+1)} = \theta^{(i)} - h \int_0^{t_f} \Big(L_\theta(\theta^{(i)}, u^{(i)}(t), \varphi^{(i)}(t)) - f_\theta(\theta^{(i)}, u^{(i)}(t), \varphi^{(i)}(t)) \eta^{(i)}(t) \Big) \, \mathrm{d}t \tag{A33}$$

with notational conventions as in Prop. 2. Eq. (A33) corresponds to a regular gradient and can be evaluated without model specific derivations based on automatic differentiation.

Alternating Gradient Descent A third alternative is to iteratively take a number of gradient steps for the u and θ while keeping the other fixed (see Algorithm 2). This method has two main advantages over the full gradient descent. First, we can use separate step sizes for the model and variational parameter updates. Second, since θ is fixed for the descent in u, we can still exploit the natural gradient for the latter. The alternating gradient descent can be seen as a hybrid between VEM and gradient descent. This is because if we performed every inner gradient descent up to convergence, the result would be equivalent to the VEM updates.

Algorithm 2 Robust Alternating Gradient Descent for Moment-Based Variational Inference

```
1: Input: Initial guesses $\theta^{(0)}$, $u^{(0)}$, initial condition $\varphi(0)$, learning rates $h_0, h_1$.
2: for $i = 0, \dots, i_{\max}$ do
3:     $u^{(i,0)} \to u^{(i)}$
4:     for $k = 0, \dots, k_{\max}$ do
5:         Set $u'$ according to (A21).
6:         if $J[\theta^{(i)}, u'] < J[\theta^{(i)}, u^{(i,k)}]$ then
7:             $h_0 \to \alpha \cdot h_0$, $u^{(i,k+1)} \to u'$
8:         else
9:             $h_0 \to \beta \cdot h_0$, $u^{(i,k+1)} \to u^{(i,k)}$
10:        end if
11:    end for
12:    $u^{(i+1)} \to u^{(i,k_{\max})}$
13:    $\theta^{(i,0)} \to \theta^{(i)}$
14:    for $k = 0, \dots, k_{\max}$ do
15:        Set $\theta'$ according to (A33).
16:        if $J[\theta', u^{(i+1)}] < J[\theta^{(i,k)}, u^{(i+1)}]$ then
17:            $h_1 \to \alpha \cdot h_1$, $\theta^{(i,k+1)} \to \theta'$
18:        else
19:            $h_1 \to \beta \cdot h_1$, $\theta^{(i,k+1)} \to \theta^{(i,k)}$
20:        end if
21:    end for
22:    $\theta^{(i+1)} \to \theta^{(i,k_{\max})}$
23: end for
24: Output: $\theta^{(i_{\max})}$, $u^{(i_{\max})}$
```

A2.5.2 Amortized Inference

In many scenarios ones observes not a single trajectory but a number of trajectories $\mathbf{x}_1, \dots, \mathbf{x}_n$ produced independently from the same model underlying model with parameter θ. Denote the corresponding noisy observations as $\mathbf{y}_1, \dots, \mathbf{y}_N$. We use boldface to indicate that $\mathbf{x}_i$, $\mathbf{y}_i$ corresponds to trajectories of stochastic processes. However, to keep the following discussion simple, we will treat them informally as ordinary random variables. Thus, the joint data likelihood is given by

$$p(\mathbf{y}_1, \dots, \mathbf{y}_n \mid \theta) = \prod_{i=1}^N p_i(\mathbf{y}_i \mid \theta) = \prod_{i=1}^N \int p_i(\mathbf{y}_i \mid \mathbf{x}_i) p_i(\mathbf{x}_i \mid \theta) \mathrm{d}\mathbf{x}_i$$

where $\mathbf{x}_i$ corresponds to the trajectory of the latent diffusion process in this case. The standard VI approach is to construct an evidence lower bound based on the proposal

$$q(\mathbf{x}_1, \dots \mathbf{x}_n) = \prod_{i=1}^{n} q_i(\mathbf{x}_i \mid u_i)\,.$$

In our case, every q_i corresponds to a full stochastic process Z_i parametrized by u_i with u_i being a function of time. Consequently, the joint variational inference problem becomes infeasible very quickly due to large memory and runtime requirements. We therefore use an amortized proposal of the form

$$q(\mathbf{x}_1, \dots \mathbf{x}_n) = \prod_{i=1}^{n} q_i(\mathbf{x}_i \mid h(y_i, \phi))$$

where h is a feed-forward neural network parametrized by ϕ. The corresponding ELBO is given by

$$\mathrm{ELBO}(\theta, \phi) = \sum_{i=1}^{N} \int q_i(\mathbf{x}_i \mid h(\mathbf{y}_i, \phi)) \log p(\mathbf{y}_i \mid \mathbf{x}_i) - D_{\mathrm{KL}}[q_i \,||\, p_i]\,. \tag{A34}$$

Now observe that every term in the sum above corresponds the objective function of the variational inference problem for a single trajectory. We may thus write

$$-\mathrm{ELBO}(\theta, \phi) = \sum_{i=1}^{N} J[\theta, h(\mathbf{y}_i, \phi)] \tag{A35}$$

with $J[\theta, u]$ as defined in Eq. A31. The key observation is now that J is a scalar function and we are able to compute gradients with respect to both θ and u. We can therefore encapsulate the computation of J and its gradients in a PyTorch module. This allows to use standard stochastic optimizers based on backpropagation to learn the model parameters θ and the inference network parameters ϕ.

A3 Experiment Details

In this section, we provide the explicit equations for the examples discussed in the main text and give more detail regarding the experiments.

Computational Resources Most experiments were run on a MacBook Pro, 2015 edition, using 2.7 GHz Intel Core i5 processor with 2 cores. We will refer to this setup as machine A. Some of the longer experiments were run on machine B; an Intel Xeon E5-2680 v3 with 2,5GHz and 22 cores. Experiments for single trajectories were generally run on machine A.

A3.1 Joint Inference and Smoothing

We test our method with a multivariate geometric Brownian motion of dimension $n = 4$. The system parameters used to generate the trajectory are given by

$$r = 10^{-4} \cdot \begin{pmatrix} 1.0 \\ 2.64 \\ 1.5 \\ 3.2 \end{pmatrix} \quad , \quad \sigma = \begin{pmatrix} 0.0112 \\ 0.0102 \\ 0.0174 \\ 0.0130 \end{pmatrix} \quad , \quad \bar{D} = \begin{pmatrix} 1 & -0.08 & -0.36 & 0.28 \\ -0.08 & 1. & 0.15 & -0.12 \\ -0.36 & 0.15 & 1. & -0.52 \\ 0.28 & -0.12 & -0.52 & 1. \end{pmatrix}.$$

Here, $\sigma = \sqrt{\mathrm{diag}(RR^\top)}$ and $\bar{D}$ represents the correlation matrix obtained from normalizing $RR^\top$ by $\sigma\sigma^\top$. The simulation was started with an initial $X_0 = (1, 1, 1, 1)^\top$. The corresponding R was obtained from D using a Cholesky decomposition. In this parameter regime, the system is noise-dominated. The parameters r are thus not identifiable and we focus on recovering the correlation structure. The systems was observed over an interval $[0, 720]$ with observations every 7 units. The observations were corrupted with Gaussian observation noise that acted independent on all components with a standard deviation $\sigma_{\mathrm{obs}} = 0.01$. As a variational process class, we used second order moments with diffusion-rescaled linear control. The required equations are given in Sec. A2.4.

Optimization was performed using Alg. 2 with $i_{\max} = 50$ and $k_{\max} = 5$. The noise matrix was initialized as $R^{(0)} = 10^{-2} \cdot I$ corresponding to a correlation free process. In the main paper, we have shown the result of a single experiment. Here, we generate $n = 100$ trajectories from the described model but use shorter trajectories over an interval $[0, 360]$. On each of these samples, we performed joint smoothing and inference. This experiment was run on machine B. We used multiprocessing with a pool of 15 workers to speed up the processing. Below, we give the average results for $\sigma*$ and $\bar{D}*$ along with corresponding standard deviation

$$\sigma^* = \begin{pmatrix} 0.0105 \pm 0.002 \\ 0.0098 \pm .001 \\ 0.0156 \pm .002 \\ 0.0118 \pm 0.001 \end{pmatrix} \quad , \quad \bar{D}^* = \begin{pmatrix} 1 & -0.03 \pm 0.15 & -0.31 \pm 0.14 & 0.23 \pm 0.14 \\ -0.03 \pm 0.15 & 1 & 0.13 \pm 0.13 & -0.08 \pm 0.14 \\ -0.31 \pm 0.14 & 0.13 \pm 0.13 & 1 & -0.46 \pm 0.11 \\ 0.23 \pm 0.14 & -0.08 \pm 0.14 & -0.46 \pm 0.11 & 1 \end{pmatrix} .$$

Although the estimates of the correlation matrix show a bit of variation, the experiments demonstrates that the inferred parameters are reasonable and fairly consistent across samples.

A3.2 Amortized Inference

We follow the approach described in Sec. A2.5.2. In order to construct the inference network, u was restricted to a piece-wise constant function on an equidistant grid. This allows to represent u as a matrix with rows and columns corresponding to the number of controls and the size of the time grid. In the specific experiment, the input layer consists of 16 unites corresponding to the 8 observations of two species. The input layer is followed by 6 ReLu-Linear layers of increasing size with the final layer matching the dimensions of the control matrix. In order to train the model, we generated 1000 trajectories using the following parametrization

$$\gamma = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.4 \end{pmatrix} , \quad \sigma = \begin{pmatrix} 0.2 & 0.1 \\ 0.1 & 0.15 \end{pmatrix} , \quad \mu = \begin{pmatrix} -1.0 \\ 1.0 \end{pmatrix} .$$

Observations were generated with independent Gaussian noise ($\sigma = 0.2$) on a time grid $t_{\text{obs}} = (2.0, 4.0, 6.0, 8.0, 10.0, 12.0, 16.0, 18.0)$. In order to speed up training, we used a mini-batch size of 15. Since gradient computation for each sample requires a forward and a backward ODE integration, we implemented the mini-batch approach using multi-processing, such that each sub-process processed one sample in the usual way.

A3.3 Code

For more details regarding hyperparameters and implementation specifics, we refer to accompanying code available at `https://git.rwth-aachen.de/bcs/projects/cw/public/mbvi_sde`.