Structured VAEs: Composing Probabilistic Graphical Models and Variational Autoencoders

Matthew J. Johnson MATTJJ@SEAS.HARVARD.EDU
Harvard University, Cambridge, MA 02138

David Duvenaud DDUVENAUD@SEAS.HARVARD.EDU
Harvard University, Cambridge, MA 02138

Alexander B. Wiltschko AWILTSCH@FAS.HARVARD.EDU
Harvard University and Twitter, Cambridge, MA 02138

Sandeep R. Datta SRDATTA@HMS.HARVARD.EDU
Harvard Medical School, Boston, MA 02115

Ryan P. Adams RPA@SEAS.HARVARD.EDU
Harvard University and Twitter, Cambridge, MA 02138

Abstract

We develop a new framework for unsupervised learning that composes probabilistic graphical models with deep learning methods and combines their respective strengths. Our method uses graphical models to express structured probability distributions and recent advances from deep learning to learn flexible feature models and bottom-up recognition networks. All components of these models are learned simultaneously using a single objective, and we develop scalable fitting algorithms that can leverage natural gradient stochastic variational inference, graphical model message passing, and backpropagation with the reparameterization trick. We illustrate this framework with a new structured time series model and an application to mouse behavioral phenotyping.

1. Introduction

Unsupervised probabilistic modeling often has two goals: first, to learn a model that is flexible enough to represent complex high-dimensional data, such as images or speech recordings, and second, to learn model structure that is interpretable, admits meaningful priors, and generalizes to new tasks. That is, it is often not enough just to learn the probability density of the data: we also want to learn a meaningful representation. Probabilistic graphical models (Koller & Friedman, 2009; Murphy, 2012) provide many tools to build such structured representations, but can be limited in their capacity and may require significant feature engineering before being applied to data. Alternatively, advances in deep learning have yielded not only flexible, scalable generative models for complex data like images but also new techniques for automatic feature learning and bottom-up inference (Kingma & Welling, 2014; Rezende et al., 2014).

Consider the problem of learning an unsupervised generative model for a depth video of a tracked freely-behaving mouse, illustrated in Figure 1. Learning interpretable representations for such data, and studying how those representations change as the animal's genetics are edited or its brain chemistry altered, can create powerful behavioral phenotyping tools for neuroscience and for high-throughput drug discovery (Wiltschko et al., 2015). Each frame of the video is a depth image of a mouse in a particular pose, and so even though each image is encoded as $30 \times 30 = 900$ pixels, the data lie near a low-dimensional nonlinear manifold. A good generative model must not only learn this manifold but also represent many other salient aspects of the data. For example, from one frame to the next the corresponding manifold points should be close to one another, and in fact the trajectory along the manifold may follow very structured dynamics. To inform the structure of these dynamics, a natural class of hypotheses used in ethology and neurobiology (Wiltschko et al., 2015) is that the mouse's behavior is composed of brief, reused actions, such as darts, rears, and grooming bouts. Therefore a natural representation would include discrete states

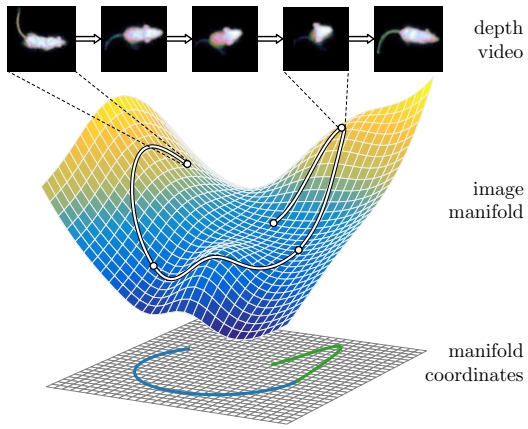


Figure 1. Illustration of unsupervised generative modeling applied to depth video of a mouse. Each video frame lies near a low-dimensional image manifold, and we wish to learn a parameterization of this manifold in which trajectories can be modeled with switching linear dynamics.

with each state representing the simple dynamics of a particular primitive action, a representation that would be difficult to encode in an unsupervised recurrent neural network model. These two tasks, of learning the image manifold and learning a structured dynamics model, are complementary: we want to learn the image manifold not just as a set but in terms of manifold coordinates in which the structured dynamics model fits the data well. A similar modeling challenge arises in speech (Hinton et al., 2012), where high-dimensional data lie near a low-dimensional manifold because they are generated by a physical system with relatively few degrees of freedom (Deng, 1999) but also include the discrete latent dynamical structure of phonemes, words, and grammar (Deng, 2004).

To address these challenges, we propose a new method to design and learn such models. Our approach uses graphical models for representing structured probability distributions, and uses ideas from variational autoencoders (Kingma & Welling, 2014) for learning not only the nonlinear feature manifold but also bottom-up recognition networks to improve inference. Thus our method enables the combination of flexible deep learning feature models with structured Bayesian and even Bayesian nonparametric priors. Our approach yields a single variational inference objective in which all components of the model are learned simultaneously. Furthermore, we develop a scalable fitting algorithm that combines several advances in efficient inference, including stochastic variational inference (Hoffman et al., 2013), graphical model message passing (Koller & Friedman, 2009), and backpropagation with the reparameterization trick (Kingma & Welling, 2014). Thus our algorithm can leverage conjugate exponential family structure where it exists to efficiently compute natural gradients with respect to some variational parameters, enabling effective second-order optimization (Martens, 2015), while using backpropagation to compute gradients with respect to all other parameters. We refer to our general approach as the structured variational autoencoder (SVAE). In this paper we illustrate the SVAE using graphical models based on switching linear dynamical systems (SLDS) (Murphy, 2012; Fox et al., 2011).

Code is available at github.com/mattjj/svae.

2. Background

Here we briefly review some of the ideas on which this paper builds. This section also fixes notation that is reused throughout the paper.

2.1. Natural gradient stochastic variational inference

Stochastic variational inference (SVI) (Hoffman et al., 2013) applies stochastic gradient ascent to a mean field variational inference objective in a way that exploits exponential family conjugacy to efficiently compute natural gradients (Amari, 1998; Martens, 2015). Consider a model composed of global latent variables θ, local latent variables $x = \{x_n\}_{n=1}^N$, and data $y = \{y_n\}_{n=1}^N$,

$$p(\theta, x, y) = p(\theta) \prod_{n=1}^{N} p(x_n|\theta) p(y_n|x_n, \theta), \tag{1}$$

where $p(\theta)$ is the natural exponential family conjugate prior to the exponential family $p(x_n, y_n|\theta)$,

$$\ln p(\theta) = \langle \eta_\theta, t_\theta(\theta) \rangle - \ln Z_\theta(\eta_\theta) \tag{2}$$

$$\begin{aligned} \ln p(x_n, y_n|\theta) &= \langle \eta_{xy}(\theta), t_{xy}(x_n, y_n) \rangle - \ln Z_{xy}(\eta_{xy}(\theta)) \\ &= \langle t_\theta(\theta), (t_{xy}(x_n, y_n),\, 1) \rangle. \end{aligned} \tag{3}$$

Figure 2 shows the graphical model. The mean field variational inference problem is to approximate the posterior $p(\theta, x \,|\, y)$ with a tractable distribution $q(\theta, x)$ by finding a local minimum of the KL divergence $\mathrm{KL}(q(\theta, x) \| p(\theta, x \,|\, y))$ or, equivalently, using the identity

$$\ln p(y) = \mathrm{KL}(q(\theta, x) \| p(\theta, x \,|\, y)) + \mathbb{E}_{q(\theta,x)} \left[\ln \frac{p(\theta, x, y)}{q(\theta, x)} \right],$$

to choose $q(\theta, x)$ to maximize the objective

$$\mathcal{L}[q(\theta, x)] \triangleq \mathbb{E}_{q(\theta,x)} \left[\ln \frac{p(\theta, x, y)}{q(\theta, x)} \right] \leq \ln p(y). \tag{4}$$

Consider the mean field family $q(\theta)q(x) = q(\theta) \prod_n q(x_n)$. Because of the conjugate exponential family structure, the optimal global mean field factor $q(\theta)$ is in the same family as the prior $p(\theta)$,

$$\ln q(\theta) = \langle \widetilde{\eta}_\theta, t_\theta(\theta) \rangle - \ln Z_\theta(\widetilde{\eta}_\theta). \tag{5}$$

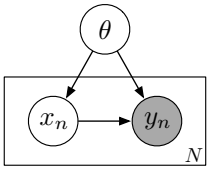
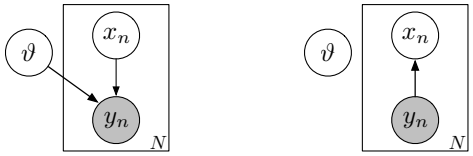


Figure 2. Prototypical graphical model for SVI.



(a) VAE generative model. (b) VAE variational family.

Figure 3. Graphical models for the variational autoencoder.

The mean field objective on the global variational parameters $\widetilde{\eta}_\theta$, optimizing out the local variational factors $q(x)$, can then be written

$$\mathcal{L}(\widetilde{\eta}_\theta) \triangleq \max_{q(x)} \mathbb{E}_{q(\theta)q(x)}\left[\ln \frac{p(\theta, x, y)}{q(\theta)q(x)}\right] \leq \ln p(y) \quad (6)$$

and the natural gradient of the objective (6) decomposes into a sum of local expected sufficient statistics (Hoffman et al., 2013):

$$\widetilde{\nabla}_{\widetilde{\eta}_\theta} \mathcal{L}(\widetilde{\eta}_\theta) = \eta_\theta + \sum_{n=1}^{N} \mathbb{E}_{q^*(x_n)}(t_{xy}(x_n, y_n),\, 1) - \widetilde{\eta}_\theta, \quad (7)$$

where $q^*(x_n)$ is a locally optimal local mean field factor given $\widetilde{\eta}_\theta$. Thus we can compute a stochastic natural gradient update for our global mean field objective by sampling a data minibatch y_n, optimizing the local mean field factor $q(x_n)$, and computing scaled expected sufficient statistics.

2.2. Variational autoencoders

The variational autoencoder (VAE) (Kingma & Welling, 2014; Rezende et al., 2014) is a recent model and variational inference method that links neural network autoencoders (Vincent et al., 2008) with mean field variational Bayes. Given a high-dimensional dataset $y = \{y_n\}_{n=1}^N$ such as a collection of images, the VAE models each observation y_n in terms of a low-dimensional latent variable x_n and a nonlinear observation model with parameters ϑ,

$$x_n \overset{\text{iid}}{\sim} \mathcal{N}(0, I), \quad n = 1, 2, \ldots, N \quad (8)$$

$$y_n \,|\, x_n, \vartheta \sim \mathcal{N}(\mu(x_n; \vartheta), \Sigma(x_n; \vartheta)) \quad (9)$$

where $\mu(x_n; \vartheta)$ and $\Sigma(x_n; \vartheta)$ might depend on x_n through a multilayer perceptron (MLP) with L layers,

$$h_\ell(x_n) = f(W_\ell h_{\ell-1}(x_n) + b_\ell),\ \ell = 1, 2, \ldots, L, \quad (10)$$

$$\mu(x_n; \vartheta) = W_\mu h_L(x_n) + b_\mu, \quad (11)$$

$$\Sigma(x_n; \vartheta) = \operatorname{diag}(\exp(W_{\sigma^2} h_L(x_n) + b_{\sigma^2})), \quad (12)$$

$$\vartheta = (\{(W_\ell, b_\ell)\}_{\ell=1}^L, (W_\mu, b_\mu), (W_{\sigma^2}, b_{\sigma^2})), \quad (13)$$

with $h_0(x_n) = x_n$ and $f(x) = \tanh(x)$ elementwise. Because we will reuse this particular MLP construction, we introduce the notation

$$(\mu(x_n; \vartheta), \Sigma(x_n; \vartheta)) = \mathrm{MLP}(x_n; \vartheta). \quad (14)$$

To approximate the posterior $p(\vartheta, x \,|\, y)$, the variational autoencoder uses the mean field family

$$q(\vartheta)q(x \,|\, y) = q(\vartheta) \prod_{n=1}^{N} q(x_n \,|\, y_n). \quad (15)$$

A key insight of the variational autoencoder is to use a conditional variational density $q(x_n \,|\, y_n)$, where the parameters of the variational distribution on x_n depend on the corresponding data point y_n. In particular, we can take the mean and covariance parameters of $q(x_n \,|\, y_n)$ to be $\mu(y_n; \phi)$ and $\Sigma(y_n; \phi)$, respectively, where

$$(\mu(y_n; \phi), \Sigma(y_n; \phi)) = \mathrm{MLP}(y_n; \phi) \quad (16)$$

and ϕ denotes a set of MLP parameters. Thus the variational distribution $q(x_n \,|\, y_n)$ acts like a stochastic *encoder* from an observation to a distribution over latent variables, while the forward model $p(y_n \,|\, x_n)$ acts as a stochastic *decoder* from a latent variable value to a distribution over observations. Furthermore, the functions $\mu(y_n; \phi)$ and $\Sigma(y_n; \phi)$ act as a *recognition* or *inference network* for observation y_n, outputting a distribution over the latent variable x_n using a feed-forward neural network applied to y_n. See Figure 3 for graphical models of the VAE.

The resulting mean field objective expresses a variational Bayesian version of an autoencoder. Using $\widetilde{\eta}_\vartheta$ to denote the variational parameters for the factor $q(\vartheta)$, the variational parameters are then the encoder parameters ϕ and the decoder parameters $\widetilde{\eta}_\vartheta$, and the objective is

$$\mathcal{L}(\widetilde{\eta}_\vartheta, \phi) \triangleq \mathbb{E}_{q(\vartheta)q(x \,|\, y)}\left[\ln \frac{p(\vartheta)}{q(\vartheta)} + \sum_{n=1}^{N} \ln \frac{p(x_n, y_n \,|\, \vartheta)}{q(x_n \,|\, y_n)}\right].$$

To optimize this objective efficiently, Kingma & Welling (2014) applies a reparameterization trick. To simplify notation and computation, we take $q(\vartheta)$ to be a singular factor $q(\vartheta) = \delta_{\vartheta^*}(\vartheta)$ so that the corresponding variational parameters $\widetilde{\eta}_\vartheta$ can be denoted ϑ^* and some terms can be dropped. First, we rewrite the objective as

$$\mathcal{L}(\vartheta^*, \phi) = \mathbb{E}_{q(x \,|\, y)} \ln p(y \,|\, x, \vartheta^*) - \mathrm{KL}(q(x \,|\, y) \,\|\, p(x)).$$

The term $\mathrm{KL}(q(x \,|\, y) \,\|\, p(x))$ is the KL divergence between two Gaussians and its gradient with respect to ϕ

can be computed in closed form. To compute stochastic gradients of the expectation term, since a random variable $x_n \sim q(x_n \mid y_n)$ can be parameterized as

$$x_n = g(\phi, \epsilon) \triangleq \mu_q(y_n; \phi) + \Sigma_q(y_n; \phi)^{\frac{1}{2}}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I),$$

the expectation term can be rewritten in terms of $g(\phi, \epsilon)$ and its gradient approximated via Monte Carlo over ϵ,

$$\nabla_{\vartheta^*,\phi}\mathbb{E}_q \ln p(y \mid x, \vartheta^*) \approx \sum_{n=1}^{N} \nabla_{\vartheta^*,\phi} \ln p(y_n \mid g(\phi, \hat{\epsilon}_n), \vartheta^*)$$

where $\hat{\epsilon}_n \overset{\text{iid}}{\sim} \mathcal{N}(0, I)$. Because $g(\phi, \epsilon)$ is a differentiable function of ϕ, these gradients can be computed using standard backpropagation. For scalability, the sum over data points is also approximated via Monte Carlo. General non-singular factors $q(\vartheta)$ can also be handled by the reparameterization trick (Kingma & Welling, 2014, Appendix F).

3. Generative model and variational family

In this section we develop an SVAE generative model and corresponding variational family. To be concrete we focus on a particular generative model for time series based on a switching linear dynamical system (SLDS) (Murphy, 2012; Fox et al., 2011), which illustrates how the SVAE can incorporate both discrete and continuous latent variables with rich probabilistic dependence. To simplify notation we consider modeling only one data sequence of length T, denoted $y = y_{1:T}$.

First, Section 3.1 describes the generative model, which illustrates the combination of a graphical model expressing latent structure with a flexible neural net to generate observations. Next, Section 3.2 describes the structured variational family, which leverages both graph-structured mean field approximations and flexible recognition networks. Section 3.3 discusses variations and extensions.

3.1. A switching linear dynamical system with nonlinear observations

A switching linear dynamical system (SLDS) represents data in terms of continuous latent states that evolve according to a discrete set of linear dynamics. At each time $t \in \{1, 2, \ldots, T\}$ there is a discrete-valued latent state $z_t \in \{1, 2, \ldots, K\}$, which indexes the dynamical mode, and a continuous-valued latent state $x_t \in \mathbb{R}^M$ that evolves according to that mode's linear Gaussian dynamics:

$$x_{t+1} = A^{(z_t)}x_t + B^{(z_t)}u_t, \quad u_t \overset{\text{iid}}{\sim} \mathcal{N}(0, I), \tag{17}$$

where $A^{(k)}, B^{(k)} \in \mathbb{R}^{M\times M}$ for $k = 1, 2, \ldots, K$. The discrete latent state z_t evolves according to Markov dynamics,

$$z_{t+1} \mid z_t, \pi \sim \pi^{(z_t)} \tag{18}$$

where $\pi = \{\pi^{(k)}\}_{k=1}^K$ denotes the Markov transition matrix and $\pi^{(k)} \in \mathbb{R}_+^n$ is its kth row. The initial states are generated separately:

$$z_1 \mid \pi_{\text{init}} \sim \pi_{\text{init}}, \tag{19}$$

$$x_1 \mid z_1, \mu_{\text{init}}, \Sigma_{\text{init}} \sim \mathcal{N}(\mu_{\text{init}}^{(z_1)}, \Sigma_{\text{init}}^{(z_1)}). \tag{20}$$

Thus inferring the latent variables and parameters of an SLDS identifies a set of reused dynamical modes, each described as a linear dynamical system on latent states, in addition to Markov switching between different linear dynamics. We use θ to denote all the dynamical parameters, $\theta = (\pi, \pi_{\text{init}}, \{(A^{(k)}, B^{(k)}, \mu_{\text{init}}^{(k)}, \Sigma_{\text{init}}^{(k)})\}_{k=1}^K)$.

At each time t, the continuous latent state x_t gives rise to an observation y_t that is conditionally Gaussian with mean $\mu(x_t; \vartheta)$ and covariance $\Sigma(x_t; \vartheta)$,

$$y_t \mid x_t, \vartheta \sim \mathcal{N}(\mu(x_t; \vartheta), \Sigma(x_t; \vartheta)). \tag{21}$$

In a typical SLDS (Fox et al., 2011), $\mu(x_t; \vartheta)$ is linear in x_t and $\Sigma(x_t; \vartheta)$ is a constant function, so that we can take $\vartheta = (C, D)$ for some matrices C and D and write

$$y_t = Cx_t + Dv_t, \quad v_t \overset{\text{iid}}{\sim} \mathcal{N}(0, I). \tag{22}$$

However, to enable flexible modeling of images and other complex features, we allow the dependence to be a more general nonlinear model. In particular, we consider $\mu(x_t; \vartheta)$ and $\Sigma(x_t; \vartheta)$ to be MLPs with parameters ϑ as in Eqs. (10)-(13) of Section 2.2,

$$(\mu(x_t; \vartheta), \Sigma(x_t; \vartheta)) = \text{MLP}(x_t; \vartheta). \tag{23}$$

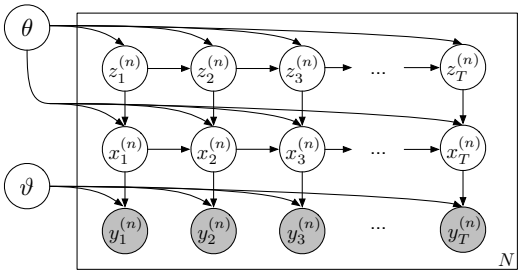
By allowing a nonlinear emission model, each low-dimensional state x_t can be mapped into a high-dimensional observation y_t, and hence the model can represent high-dimensional data in terms of structured dynamics on a low-dimensional manifold. The overall generative model is summarized in Figure 4a.

Note that by construction the density $p(z, x \mid \theta)$ is an exponential family. We can choose the prior $p(\theta)$ to be a natural exponential family conjugate prior, writing

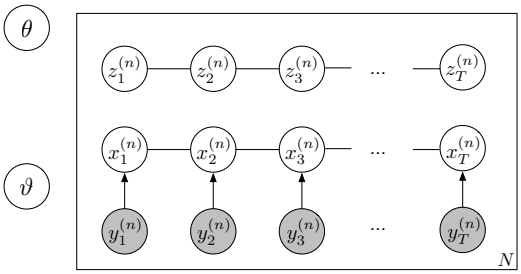
$$\ln p(\theta) = \langle \eta_\theta, t_\theta(\theta)\rangle - \ln Z_\theta(\eta_\theta) \tag{24}$$

$$\begin{aligned}\ln p(z, x|\theta) &= \langle \eta_{zx}(\theta), t_{zx}(z, x)\rangle - \ln Z_{zx}(\eta_{zx}(\theta)) \\ &= \langle t_\theta(\theta), (t_{zx}(z, x),\, 1)\rangle.\end{aligned} \tag{25}$$

We can also use a Bayesian nonparametric prior, choosing $K = \infty$ and generating the discrete state sequence $z_{1:T}$ according to an HDP-HMM (Fox et al., 2011). Though we do not discuss the Bayesian nonparametric case further, the algorithms we develop here immediately extend to the HDP-HMM using the methods in Johnson & Willsky (2014).

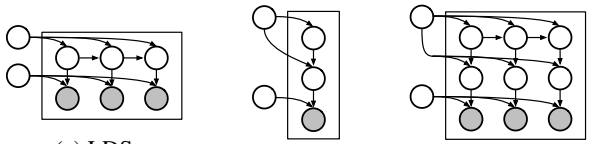


(a) SLDS generative model with nonlinear observation model parameterized by ϑ.



(b) Structured CRF variational family with node potentials $\{\psi(x_t^{(n)}; y_t^{(n)}, \phi)\}_{t=1}^T$ parameterized by ϕ.

Figure 4. Graphical models for the SLDS generative model and corresponding structured CRF variational family.



(a) LDS (b) GMM (c) G-HMM

Figure 5. Special cases of the SLDS generative model.

This construction contains the generative model of the VAE, described in Section 2.2, as a special case. Specifically, the VAE uses the same class of MLP observation models, but each latent value x_t is modeled as an independent and identically distributed Gaussian, $x_t \overset{\text{iid}}{\sim} \mathcal{N}(0, I)$, while the SVAE model proposed here allows the x_t to have a rich joint probabilistic structure. The SLDS generative model also includes as special cases the Gaussian mixture model (GMM), Gaussian-emission discrete-state HMM (G-HMM), and Gaussian linear dynamical system (LDS), and thus the algorithms we develop here for the SLDS directly specialize to these models. See Figure 5 for graphical models of these special case models.

While using conditionally linear dynamics within each state may seem limited, the flexible nonlinear observation distribution greatly extends the capacity of such models. Indeed, recent work on neural word embeddings (Mikolov et al., 2013) as well as neural image models (Radford et al., 2015) has demonstrated learned latent spaces in which linear structure corresponds to meaningful semantics. For example, addition and subtraction of word vectors can correspond to semantic relationships between words and translation in an image model's latent space can correspond to an object's rotation. Therefore linear models in a learned latent space can yield significant expressiveness while enabling fast probabilistic inference, interpretable priors and parameters, and a host of other tools. In particular, linear dynamics allow us to learn or encode information about timescales and frequencies: the eigenvalue spectrum of each transition matrix $A^{(k)}$ directly represents its characteristic timescales, and so we can control and interpret the structure of linear dynamics in ways that nonlinear dynamics models do not allow.

3.2. Variational family and CRF recognition networks

Here we describe a structured mean field family with which we can perform variational inference in the posterior distribution of the generative model from Section 3.1. This mean field family illustrates how an SVAE can leverage not only graphical model and exponential family structure but also learn bottom-up inference networks. As we show in Section 4, these structures allow us to compose several efficient inference algorithms including SVI, message passing, backpropagation, and the reparameterization trick.

In mean field variational inference, one constructs a tractable variational family by breaking dependencies in the posterior (Wainwright & Jordan, 2008). To construct a structured mean field family for the generative model developed in Section 3.1, we break the posterior dependencies between the dynamics parameters θ, the observation parameters ϑ, the discrete state sequence $z = z_{1:T}$ and the continuous state sequence $x = x_{1:T}$, writing the corresponding a factorized density as

$$q(\theta, \vartheta, z_{1:T}, x_{1:T}) = q(\theta)q(\vartheta)q(z_{1:T})q(x_{1:T}). \tag{26}$$

Note that this structured mean field family does not break the dependencies among the discrete states $z_{1:T}$ or among the continuous states $x_{1:T}$ as in a naive mean field model because these random variables are highly correlated in the posterior. By preserving joint dependencies across time, these structured factors provide a much more accurate representation of the posterior while still allowing tractable inference via graphical model message passing (Wainwright & Jordan, 2008).

To leverage bottom-up inference networks, we parameterize the factor $q(x_{1:T})$ as a conditional random field (CRF) (Murphy, 2012). That is, using the fact that the optimal factor $q(x_{1:T})$ is Markov according to a chain graph, we write

it terms of pairwise potentials and node potentials as

$$q(x_{1:T}) \propto \left(\prod_{t=1}^{T-1} \psi(x_t, x_{t+1})\right)\left(\prod_{t=1}^{T} \psi(x_t; y_t, \phi)\right) \quad (27)$$

where the node potential $\psi(x_t; y_t, \phi)$ is a function of the observation y_t. Specifically, we choose each node potential to be a Gaussian factor in which the precision matrix $J(y_t; \phi)$ and potential vector $h(y_t; \phi)$ depend on the corresponding observation y_t through an MLP,

$$\psi(x_t; y_t, \phi) \propto \exp\left\{-\frac{1}{2}x_t^\mathsf{T} J(y_t; \phi)x_t + h(y_t; \phi)^\mathsf{T} x_t\right\},$$
$$(h(y_t; \phi), J(y_t; \phi)) = \mathrm{MLP}(y_t; \phi), \quad (28)$$

using the notation from Section 2.2. These *local recognition networks* allow us to fit a regression from each observation y_t to a probabilistic guess at the corresponding latent state x_t. Using graphical model inference, these local guesses can be synthesized with the dynamics model into a coherent joint factor $q(x_{1:T})$ over the entire state sequence. While we could write $q(x_{1:T} \mid y_{1:T})$ to emphasize this dependence and match the notation in Section 2.2, we only write $q(x_{1:T})$ to simplify notation. The overall variational family is summarized in Figure 4b.

There are many alternative designs for such recognition networks; for example, the node potential on x_t need not depend only on the observation y_t. See Section 3.3.

This structured mean field family can be directly compared to the fully factorized family used in the variational autoencoder described in Section 2.2. That is, there is no graph structure among the latent variables of the VAE. The SVAE generalizes the VAE by allowing the output of the recognition network to be arbitrary potentials in a graphical model, such as the node potentials considered here. Furthermore, in the SVAE some of the graphical model potentials are induced by the probabilistic model rather than being the output of a recognition network; for example, the optimal pairwise potentials $\psi(x_t, x_{t+1})$ are induced by the variational factors on the dynamical parameters and latent discrete states, $q(\theta)$ and $q(z_{1:T})$, and the forward generative model $p(z, x \mid \theta)$ (see Section 4.2.1). Thus the SVAE provides a way to combine bottom-up information from flexible inference networks with top-down information from other latent variables in a structured probabilistic model.

When $p(\theta)$ is chosen to be a conjugate prior, as in Eq. (24), the optimal factor $q(\theta)$ is in the same exponential family:

$$q(\theta) = \exp\left\{\langle \widetilde{\eta}_\theta, t_\theta(\theta)\rangle - \log Z_\theta(\widetilde{\eta}_\theta)\right\}, \quad (29)$$

where $\widetilde{\eta}_\theta$ denotes the variational natural parameters. To simplify notation, as in Section 2.2 we take the variational factor on the observation parameters to be a singular distribution, $q(\vartheta) = \delta_{\vartheta^*}(\vartheta)$. The mean field objective in terms

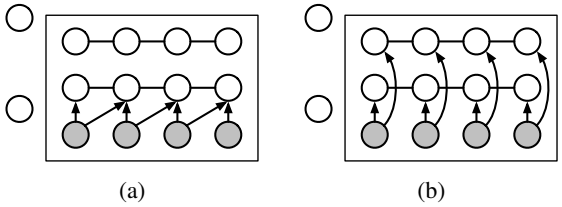


Figure 6. Alternative SLDS recognition models.

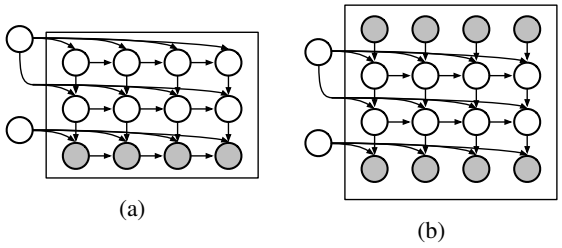


Figure 7. Variations on the SLDS generative model.

of the global variational parameters $\widetilde{\eta}_\theta$, ϑ^*, and ϕ is then

$$\mathcal{L}(\widetilde{\eta}_\theta, \vartheta^*, \phi) \triangleq \max_{q(z)q(x)} \mathbb{E}_{q(\theta)q(z)q(x)}\left[\ln \frac{p(\theta, z, x)p(y \mid x, \vartheta^*)}{q(\theta)q(z)q(x)}\right] \quad (30)$$

where, as in Eq. (6), the maximization is over the free parameters of the local variational factors $q(z)$ and $q(x)$ given fixed values of $\widetilde{\eta}_\theta$, ϑ^*, and ϕ. In Section 4 we show how to optimize this variational objective.

3.3. Variations and extensions

The SVAE construction is general: it can admit many latent probabilistic models as well as many flexible observation models and recognition network designs. In this section we outline some of these possibilities.

While the SVAE recognition network described in Section 3.2 only produces node potentials depending on single data points, in general SVAE recognition networks can output potentials on more than one node or take as input more than one data point. For example, a recognition network could output node potentials of the form $\phi(x_t; y_t, y_{t-1})$ that depend also on the previous data point, as sketched in Figure 6a, or even depend on many data points through a recurrent neural network (RNN). Recognition networks may also output factors on discrete latent variables, as sketched in Figure 6b.

The observation model may also be extended, for example to allow data to be generated according to a nonlinear autoregression, as sketched in Figure 7a. This extension may be useful when modeling video so that a frame can be generated in part by reference to the preceding frame.

Finally, the SVAE also admits many generative models,

both structured probabilistic graphical models and more flexible alternatives. As mentioned in Section 3.2, the SLDS described here includes many common structured graphical models as special cases, shown in Figure 5. For example, by using a GMM inside an SVAE, one can express a warped mixture model (Iwata et al., 2013). The SLDS also lends itself to many extensions, including explicit-duration semi-Markov modeling of discrete states (Johnson & Willsky, 2013) or IO-HMMs (Bengio & Frasconi, 1995), sketched in Figure 7b. More generally, SVAEs provide a direct way to incorporate hierarchical Bayesian modeling, which enables many ways to share latent random variables and generalize across datasets. Because the SVAE can draw on the vast literature on probabilistic graphical models, there is a rich library of potential SVAEs. Finally, though we focus on traditional probabilistic graphical models here, the SVAE generative model can also be an RNN expressing flexible nonlinear dynamics (Archer et al., 2015), though such models would not in general admit the efficient inference algorithms and natural gradients provided by exponential families.

4. Learning and inference

In this section we give an efficient algorithm for computing stochastic gradients of the SVAE objective in Eq. (30). These stochastic gradients can be used in a generic optimization routine such as stochastic gradient ascent or Adam (Kingma & Ba, 2015).

As we show, the SVAE algorithm is essentially a combination of SVI (Hoffman et al., 2013) and AEVB (Kingma & Welling, 2014), described in Sections 2.1 and 2.2, respectively. By drawing on SVI, our algorithm is able to exploit exponential family conjugacy structure, when it is available, to efficiently compute natural gradients with respect to some variational parameters. Because natural gradients are adapted to the geometry of the variational family and are invariant to model reparameterizations (Amari & Nagaoka, 2007) natural gradient ascent provides an effective second-order optimization method (Martens & Grosse, 2015; Martens, 2015). By drawing on AEVB, our algorithm can fit both general nonlinear observation models and flexible bottom-up recognition networks.

The algorithm is split into two parts. First, in Section 4.1 we describe the general algorithm for computing gradients of the objective in terms of the results from a model-specific inference subroutine. Next, in Section 4.2 we detail this model inference subroutine for the SLDS.

4.1. SVAE algorithm

Here we show how to compute stochastic gradients of the SVAE mean field objective (30) using the results of a model inference subroutine. The algorithm is summarized in Algorithm 1 and implemented in svae.py.

For scalability, the stochastic gradients used here are computed on minibatches of data. To simplify notation, we assume the dataset is a collection of N sequences, $\{y^{(n)}\}_{n=1}^N$, each of length T. We sample one sequence $y^{(n)}$ uniformly at random and compute a stochastic gradient with it. It is also possible to sample subsequences and compute controllably-biased stochastic gradients (Foti et al., 2014).

The SVAE algorithm computes a natural gradient with respect to $\widetilde{\eta}_\theta$ and standard gradients with respect to ϑ^* and ϕ. To compute these gradients, as in Section 2.2 we split the objective $\mathcal{L}(\widetilde{\eta}_\theta, \vartheta^*, \phi)$ as

$$\mathbb{E}_{q(x)} \ln p(y \mid x, \vartheta^*) - \mathrm{KL}(q(\theta, z, x) \,\|\, p(\theta, z, x)). \quad (31)$$

Note that only the second term depends on the variational dynamics parameter $\widetilde{\eta}_\theta$. Furthermore, it is a KL divergence between two members of the same exponential family (Eqs. (24) and (29)), and so as in Hoffman et al. (2013) and Section 2.1 we can write the natural gradient of (31) with respect to $\widetilde{\eta}_\theta$ as

$$\widetilde{\nabla}_{\widetilde{\eta}_\theta} \mathcal{L} = \eta_\theta + \sum_{n=1}^N \mathbb{E}_{q(z)q(x)}(t_{zx}(z^{(n)}, x^{(n)}),\, 1) - \widetilde{\eta}_\theta \quad (32)$$

where $q(z)$ and $q(x)$ are taken to be locally optimal local mean field factors as in Eq. (7). Therefore by sampling the sequence index n uniformly at random, an unbiased estimate of the natural gradient is given by

$$\widetilde{\nabla}_{\widetilde{\eta}_\theta} \mathcal{L} \approx \eta_\theta + N\mathbb{E}_{q(z)q(x)}(t_{zx}(z^{(n)}, x^{(n)}),\, 1) - \widetilde{\eta}_\theta. \quad (33)$$

We abbreviate $\bar{t}_{zx}^{(n)} \triangleq \mathbb{E}_{q(z)q(x)} t_{zx}(z^{(n)}, x^{(n)})$. These expected sufficient statistics are computed efficiently using the model inference subroutine described in Section 4.2.

To compute the gradient of the SVAE objective with respect to the observation parameters ϑ^*, note that only the first term in (31) depends on ϑ^*. As in Section 2.2, we can compute a Monte Carlo approximation to this gradient using samples $\{\hat{x}_s^{(n)}(\phi)\}_{s=1}^S$ where $\hat{x}_s^{(n)}(\phi) \overset{\text{iid}}{\sim} q(x^{(n)})$ and we write $\hat{x}_s^{(n)}(\phi)$ to note the dependence on ϕ. These samples are generated efficiently (and as differentiable functions of ϕ) by the model inference subroutine described in Section 4.2. To simplify notation we take $S = 1$.

Finally, we compute gradients with respect to the recognition network parameters ϕ. Both the first and second terms of (31) depend on ϕ, the first term through the sample $\hat{x}^{(n)}(\phi)$ and the second term through the KL divergence $\mathrm{KL}(\phi) \triangleq \mathrm{KL}(q(\theta, z^{(n)}, x^{(n)}) \,\|\, p(\theta, z^{(n)}, x^{(n)}))$. Thus we must differentiate through the procedures that the model inference subroutine uses to compute these quantities. As

Algorithm 1 Computing gradients of the SVAE objective

Input: Variational dynamics parameter $\widetilde{\eta}_\theta$ of $q(\theta)$, observation model parameter ϑ^*, recognition network parameters ϕ, sampled sequence $y^{(n)}$
function SVAEGRADIENTS($\widetilde{\eta}_\theta$, ϑ^*, ϕ, $y^{(n)}$)
$\psi \leftarrow \text{RECOGNIZE}(y^{(n)}, \phi)$
$(\hat{x}^{(n)}(\phi),\ \bar{t}_{zx}^{(n)},\ \mathrm{KL}(\phi)) \leftarrow \text{INFERENCE}(\widetilde{\eta}_\theta, \psi)$
$\widetilde{\nabla}_{\widetilde{\eta}_\theta}\mathcal{L} \leftarrow \eta_\theta + N(\bar{t}_{zx}^{(n)},\ 1) - \widetilde{\eta}_\theta$
$\nabla_{\vartheta^*,\phi}\mathcal{L} \leftarrow \nabla_{\vartheta^*,\phi}\big[N \ln p(y^{(n)} | \hat{x}^{(n)}(\phi), \vartheta^*) - \mathrm{KL}(\phi)\big]$
returnnatural gradient $\widetilde{\nabla}_{\widetilde{\eta}_\theta}\mathcal{L}$, gradient $\nabla_{\vartheta^*,\phi}\mathcal{L}$
end function

described in Section 4.2, performing this differentiation efficiently for the SLDS corresponds to backpropagation through message passing.

To observe that both $\hat{x}^{(n)}(\phi)$ and $\mathrm{KL}(\phi)$ can be computed differentiably by the model inference subroutine of the SLDS, and to relate this process to the reparameterization trick of Section 2.2, note that the objective only depends on ϕ through the optimal factor $q(x_{1:T}^{(n)})$, which is a joint Gaussian factor over the latent continuous states. That is, we can write the sample $\hat{x}^{(n)}(\phi) \sim q(x_{1:T}^{(n)})$ as

$$\hat{x}^{(n)}(\phi) = g(\phi, \epsilon) \triangleq J^{-1}(\phi)h(\phi) + J(\phi)^{-\frac{1}{2}}\epsilon \qquad (34)$$

where $\epsilon \sim \mathcal{N}(0, I)$, $h(\phi) \in \mathbb{R}^{TM}$, and $J(\phi) \in \mathbb{R}^{TM \times TM}$. The matrix $J(\phi)$ is a block tridiagonal matrix corresponding to the Gaussian LDS of (27), the block diagonal of which depends on ϕ. Since $g(\phi, \epsilon)$ is differentiable with respect to ϕ, this representation shows that $\hat{x}^{(n)}(\phi)$ is differentiable with respect to ϕ. Given a sample $\hat{\epsilon}$, to evaluate $\nabla_\phi g(\phi, \hat{\epsilon})$ efficiently we can exploit the block tridiagonal structure of $J(\phi)$, which corresponds exactly to backpropagating through the message passing procedure used to sample $q(x_{1:T}^{(n)})$. Similarly, to evaluate the gradient of the second term of (31), which is computed analytically, we simply backpropagate through the message passing routine used to compute it. Both message passing computations are part of the model inference subroutine described in Section 4.2. Computing each of these gradients using standard backpropagation tools requires twice the time of the message passing algorithms used to compute them. When the reparameterization trick is not available, the score function estimator can be used instead (Kleijnen & Rubinstein, 1996; Gelman & Meng, 1998; Ranganath et al., 2014)

4.2. Model inference subroutine

Here we describe the model inference subroutine used by the SVAE algorithm.

Because the VAE corresponds to a particular SVAE with limited latent probabilistic structure, this inference subroutine can be viewed as a generalization of two steps in the AEVB algorithm (Kingma & Welling, 2014). Specifically, using the notation of Section 2.2, in the special case of the VAE the inference subroutine computes $\mathrm{KL}(q(x \mid y) \,\|\, p(x))$ in closed form and generates samples $x \sim q(x \mid y)$ used to approximate $\mathbb{E}_{q(x)} \ln p(y \mid x)$. However, the inference subroutine of an SVAE may in general perform other computations: first, because the SVAE can include other latent random variables and graph structure, the inference subroutine may optimize local mean field factors or run message passing. Second, because the SVAE can perform stochastic natural gradient updates on the global factor $q(\theta)$, the inference subroutine may also compute expected sufficient statistics.

Algorithm 2 Model inference subroutine for the SLDS

Input: Variational dynamics parameter $\widetilde{\eta}_\theta$ of $q(\theta)$, node potentials $\{\psi(x_t; y_t)\}_{t=1}^T$ from recognition network
function INFERENCE($\widetilde{\eta}_\theta$, $\{\psi(x_t; y_t)\}_{t=1}^T$)
Initialize factor $q(x)$
repeat
$q(z) \propto \exp\{\mathbb{E}_{q(\theta)q(x)} \ln p(z, x \mid \theta)\}$
$q(x) \propto \exp\{\mathbb{E}_{q(\theta)q(z)} \ln p(x \mid z, \theta)\} \prod_t \psi(x_t; y_t)$
until $q(z)$ and $q(x)$ converge
$\hat{x} \leftarrow$ sample $q(x)$
$\bar{t}_{zx} \leftarrow \mathbb{E}_{q(z)q(x)} t_{zx}(z, x)$
$\mathrm{KL} \leftarrow \mathrm{KL}(q(\theta) \,\|\, p(\theta))$
$+N\mathbb{E}_{q(\theta)} \mathrm{KL}(q(z)q(x) \,\|\, p(z, x \mid \theta))$
returnsample $\hat{x}$, expected stats $\bar{t}_{zx}$, divergence KL
end function

In this subsection, we detail these computations for the SLDS model. Specifically, the inference subroutine must first optimize the local mean field factors $q(z_{1:T})$ and $q(x_{1:T})$, then compute and return a sample $\hat{x}_{1:T} \sim q(x_{1:T})$, expected sufficient statistics $\mathbb{E}_{q(z)q(x)} t_{zx}(z, x)$, and a KL divergence $\mathrm{KL}(q(\theta, z, x) \,\|\, p(\theta, z, x))$. To simplify notation, we drop the sequence index n, writing y in place of $y^{(n)}$. The algorithm is summarized in Algorithm 2 and implemented in slds_svae.py.

4.2.1. Optimizing local mean field factors

As in the SVI algorithm of Section 2.1, for a given data sequence y we need to optimize the local mean field factors $q(z)$ and $q(x)$. That is, for a fixed global parameter factor $q(\theta)$ with natural parameter $\widetilde{\eta}_\theta$ and fixed node potentials $\psi = \{\psi(x_t; y_t)\}_{t=1}^T$ output by the recognition network, we optimize the variational objective with respect to both the local variational factor on discrete latent states $q(z_{1:T})$ and the local variational factor on continuous latent states $q(x_{1:T})$. This optimization can be performed efficiently by exploiting the SLDS exponential family form and the structured variational family.

The algorithm proceeds by alternating between optimizing $q(x_{1:T})$ and optimizing $q(z_{1:T})$. Fixing the factor on the discrete states $q(z_{1:T})$, the optimal variational factor on the continuous states $q(x_{1:T})$ is proportional to

$$\exp\left\{\ln\psi(x_1) + \sum_{t=1}^{T-1}\ln\psi(x_t, x_{t+1}) + \sum_{t=1}^{T}\ln\psi(x_t; y_t)\right\},$$

where

$$\psi(x_1) = \mathbb{E}_{q(\theta)q(z)}\ln p(x_1 \mid z_1, \theta) \tag{35}$$
$$\psi(x_t, x_{t+1}) = \mathbb{E}_{q(\theta)q(z)}\ln p(x_{t+1} \mid x_t, z_t, \theta), \tag{36}$$

Because the densities $p(x_1 \mid z_1, \theta)$ and $p(x_{t+1} \mid x_t, z_t, \theta)$ are Gaussian exponential families, the expectations in Eqs. (35)-(36) can be computed efficiently, yielding Gaussian potentials with natural parameters that depend on both $q(\theta)$ and $q(z_{1:T})$. Furthermore, because each $\psi(x_t; y_t)$ is itself a Gaussian potential, the overall factor $q(x_{1:T})$ is a Gaussian linear dynamical system with natural parameters computed from the variational factor on the dynamical parameters $q(\theta)$, the variational parameter on the discrete states $q(z_{1:T})$ and the recognition model potentials $\{\psi(x_t; y_t)\}_{t=1}^{T}$.

Because the optimal factor $q(x_{1:T})$ is a Gaussian linear dynamical system, we can use message passing to perform efficient inference. In particular, the expected sufficient statistics of $q(x_{1:T})$ needed for updating $q(z_{1:T})$ can be computed efficiently. Message passing can also be used to draw samples or compute the log partition function efficiently, as we describe in Section 4.2.2.

Similarly, fixing $q(x_{1:T})$ the optimal factor $q(z_{1:T})$ is proportional to

$$\exp\left\{\ln\psi(z_1) + \sum_{t=1}^{T-1}\ln\psi(z_t, z_{t+1}) + \sum_{t=1}^{T}\ln\psi(z_t)\right\},$$

where

$$\begin{aligned}\psi(z_1) &= \mathbb{E}_{q(\theta)}\ln p(z_1 \mid \theta) + \mathbb{E}_{q(\theta)q(x)}\ln p(x_1 \mid z_1, \theta)\\ \psi(z_t, z_{t+1}) &= \mathbb{E}_{q(\theta)}\ln p(z_{t+1} \mid z_t)\\ \psi(z_t) &= \mathbb{E}_{q(\theta)q(x)}\ln p(x_{t+1} \mid x_t, z_t, \theta)\end{aligned} \tag{37}$$

Again, because the densities involved are exponential families, these expectations can be computed efficiently. The resulting factor $q(z_{1:T})$ is an HMM with natural parameters that are functions of $q(\theta)$ and $q(x_{1:T})$. Both the expected sufficient statistics required for updating $q(x_{1:T})$ as well as the log partition function required in Section 4.2.2 can be computed efficiently by message passing.

4.2.2. Samples, expected statistics, and KL

After optimizing the local variational factors, the model inference subroutine uses the optimized factors to draw samples, compute expected sufficient statistics, and compute a KL divergence. The results of these inference computations, which we detail here, are then used to compute gradients of the SVAE objective as described in Section 4.1.

To draw S samples $\{\hat{x}^{(s)}\}_{s=1}^{S}$ where $\hat{x}_{1:T}^{(s)} \overset{\text{iid}}{\sim} q(x_{1:T})$, we can perform message passing in $q(x_{1:T})$, which is a Gaussian distribution that is Markov with respect to a chain graph. Given two neighboring nodes i and j we define the message from i to j as

$$m_{i\to j}(x_j) \triangleq \int \psi(x_i; y_i)\psi(x_i, x_j)m_{k\to i}(x_i)\,\mathrm{d}x_i \tag{38}$$

where k is the other neighbor of node i. We can pass messages backward once, running a Kalman filter, and then sample forward S times. That is, after computing the messages $m_{t+1\to t}(x_t)$ for $t = T-1, \ldots, 1$, we compute the marginal distribution on x_1 as

$$q(x_1) \propto \psi(x_1)\psi(x_1; y_1)m_{2\to 1}(x_1) \tag{39}$$

and sample $\hat{x}_1 \sim q(x_1)$. Iterating, given a sample of $\hat{x}_{t-1}$, we sample $\hat{x}_t$ from the conditional distributions

$$\begin{aligned}q(x_t \mid \hat{x}_{1:t-1}) &\propto \psi(\hat{x}_{t-1}, x_t)\psi(x_t; y_t)m_{t+1\to t}(x_t),\\ q(x_T \mid \hat{x}_{1:T-1}) &\propto \psi(\hat{x}_{T-1}, x_T)\psi(x_T; y_T),\end{aligned} \tag{40}$$

thus constructing a joint sample $\hat{x}_{1:T} \sim q(x_{1:T})$.

To compute the expected sufficient statistics for the natural gradient step on $\widetilde{\eta}_\theta$ we can also use message passing, this time in both factors $q(x_{1:T})$ and $q(z_{1:T})$ separately. Recall the exponential family notation from Eqs. (24)-(25). The required expected sufficient statistics, which we abbreviate as $\bar{t}_{zx} \triangleq \mathbb{E}_{q(x)q(z)}t_{zx}(z, x)$, are of the form

$$\begin{aligned}&\mathbb{E}_{q(z)}\mathbb{I}[z_t = i, z_{t+1} = j], \quad \mathbb{E}_{q(z)}\mathbb{I}[z_t = i],\\ &\mathbb{E}_{q(z)}\mathbb{I}[z_t = k]\mathbb{E}_{q(x)}\left[x_t x_{t+1}^{\mathsf{T}}\right],\\ &\mathbb{E}_{q(z)}\mathbb{I}[z_t = k]\mathbb{E}_{q(x)}\left[x_t x_t^{\mathsf{T}}\right], \quad \mathbb{E}_{q(z)}\mathbb{I}[z_1 = k]\mathbb{E}_{q(x)}[x_1],\end{aligned} \tag{41}$$

where $\mathbb{I}[\cdot]$ denotes an indicator function. Each of these can be computed easily from the marginals $q(x_t, x_{t+1})$ and $q(z_t, z_{t+1})$ for $t = 1, 2, \ldots, T-1$, and these marginals can be computed in terms of the respective graphical model messages. For example, to compute the marginal $q(x_t, x_{t+1})$, we write

$$\begin{aligned}q(x_t, x_{t+1}) \propto\ &\psi(x_t; y_t)\psi(x_t, x_{t+1})\psi(x_{t+1}; y_{t+1})\\ &\cdot m_{t-1\to t}(x_t)m_{t+2\to t+1}(x_{t+1}).\end{aligned} \tag{42}$$

Because each of the factors is a Gaussian potential, the product yields a Gaussian marginal with natural parameters computed from the sum of the factors' natural parameters. The corresponding expected sufficient statistics can then be computed by converting from natural parameters to mean parameters. The computations for $q(z_t, z_{t+1})$ are similar.

Finally, to compute the required KL divergence efficiently, we split $\mathrm{KL}(q(\theta, z, x) \,\|\, p(\theta, z, x))$ into three terms:

$$\mathrm{KL}(q(\theta, z, x) \,\|\, p(\theta, z, x)) = \mathrm{KL}(q(\theta) \,\|\, p(\theta)) + \mathbb{E}_{q(\theta)} \mathrm{KL}(q(z) \| p(z \,|\, \theta)) + \mathbb{E}_{q(\theta)q(z)} \mathrm{KL}(q(x) \| p(x \,|\, z, \theta)). \tag{43}$$

Each term is a divergence between members of the same exponential family and thus can be computed in terms of natural parameters, expected sufficient statistics, and log partition functions. In particular, the first term is

$$\mathbb{E}_{q(\theta)}\left[\ln \frac{q(\theta)}{p(\theta)}\right] = \langle \widetilde{\eta}_\theta - \eta_\theta,\ \bar{t}_\theta \rangle - (\ln Z_\theta(\widetilde{\eta}_\theta) - \ln Z_\theta(\eta_\theta)), \qquad \bar{t}_\theta \triangleq \mathbb{E}_{q(\theta)} t_\theta(\theta). \tag{44}$$

Because the factors $q(z)$ and $q(x)$ are locally optimal for $q(\theta)$ (Beal, 2003), the second and third terms are simply the log partition functions of $q(z)$ and $q(x)$, respectively, each of which can be computed efficiently from message passing, along with some linear terms. Note that the third term in the KL divergence (43) is a function of the recognition network parameters ϕ through the conditional node potentials $\{\psi(x_t; y_t, \phi)\}_{t=1}^T$ in $q(x)$.

5. Related work

In addition to the papers already referenced, there are several recent papers to which this work is related. The two papers closest to this work are Krishnan et al. (2015) and Archer et al. (2015).

In Krishnan et al. (2015) the authors consider combining variational autoencoders with continuous state-space models, emphasizing the relationship to linear dynamical systems (also called Kalman filter models). They primarily focus on nonlinear dynamics and an RNN-based variational family, as well as allowing control inputs (analogous to the IO-HMM extension mentioned in Section 3.3 and in Figure 7b, though with no discrete states and with the covariates influencing the continuous states). The approach in Krishnan et al. (2015) covers many nonlinear probabilistic state-space models but does not extend to general graphical models and in particular discrete latent variables. That approach also does not leverage SVI, natural gradients, or efficient message passing inference in the case of (switching) linear dynamics. However, their nonlinear dynamcis model is more general than the conditionally linear dynamics we explicitly consider here. The authors also emphasize causal and counterfactual inference.

In Archer et al. (2015) the authors also consider the problem of variational inference in general continuous state space models but focus on using a structured Gaussian variational family. They show how to extend the AEVB algorithm of Kingma & Welling (2014) to leverage block tridiagonal-structured precision (inverse covariance) matrices, and hence develop a method more suitable for time series and state space models. Leveraging this tridiagonal structure in the AEVB algorithm is the same as the backpropagation through message passing in $q(x_{1:T})$ discussed in Section 4.2.2, though our algorithm also backpropagates through message passing in the discrete state factor $q(z_{1:T})$. The authors do not discuss learning dynamics models; instead, they focus on the case of inference with known nonlinear dynamics. As with Krishnan et al. (2015), their model does not include discrete latent variables (or any latent variables other than the continuous states) and does not discuss SVI for learning. The overall algorithm of Archer et al. (2015) can be most directly compared to part of the model inference subroutine we discuss in Section 4.2, namely to the steps of optimizing the chain-structured factor $q(x_{1:T})$ and performing efficient inference in it, with the key difference of emphasizing nonlinear dynamics. Indeed, the inference algorithm of Archer et al. (2015) can be used in an SVAE to handle inference with conditionally nonlinear dynamics, and so these works are complementary. Notably, Section 4.1 and specifically Eq. (16) of Krishnan et al. (2015) demonstrate a recognition network that outputs pairwise potentials (i.e. dynamics potentials).

There are many other recent related works, especially leveraging new deep learning ideas. In particular, both Gregor et al. (2015) and Chung et al. (2015) extend the variational autoencoder framework to sequential models, though they focus on models based on RNNs rather than probabilistic graphical models. More classical approaches for nonlinear filtering, smoothing, and model fitting are surveyed in Krishnan et al. (2015).

6. Experiments

In this section we apply the SVAE to both synthetic and real data and demonstrate its ability to learn both rich feature representations and simple latent dynamics. First, we apply a linear dynamical system (LDS) SVAE to synthetic data and illustrate some aspects of its training dynamics. Second, we apply an LDS SVAE and switching linear dynamical system (SLDS) SVAE to modeling depth video recordings of mouse behavior.

6.1. Image of a moving dot

As a synthetic data example, consider a sequence of 1D images representing a dot bouncing from one side of the image to the other, as shown in the top panel of Figure 8a. While these images are simple, the task of modeling such sequences captures many salient aspects of the SVAE: to make predictions far into the future with coherent uncertainty estimates, we can use an LDS SVAE to find a low-

dimensional latent state space representation, along with a nonlinear image model and a simple model of dynamics in the latent state space.

We trained the LDS SVAE on 80 random image sequences each of length 50, using one sequence per update, and show the model's future predictions given a prefix of a longer sequence. We used MLP image and recognition models each with one hidden layer of 50 units. Figures 8a and 8b show predictions from an LDS SVAE at two stages of training. Taken from an early stage in training, Figure 8a shows that the LDS SVAE first learns an autoencoder that can represent the image data accurately. However, while the latent space allows the images to be encoded and decoded accurately, the latent space trajectories are not well modeled by a linear dynamical system and hence future predictions are highly uncertain. As training proceeds, the SVAE adjusts the latent space representation so that the trajectories can be accurately modeled and predicted using an LDS, as shown in Figure 8b, and hence long-range predictions become very accurate. Figure 9 shows the same experiment repeated with a higher resolution image.

Figure 10 shows predictions from the fit LDS SVAE based on revealing data prefixes of different lengths. Note that when only a single frame is revealed, the model is uncertain as to whether the dot is moving upwards or downwards in the image. However, instead of showing a bifurcation in the predictions corresponding to these two distinct hypotheses, because the latent dynamics are linear the model's predictions encompass all possible trajectories between these two hypotheses. With a nonlinear dynamics model, such as a switching LDS (SLDS) or a more general recurrent neural network (RNN), a model that explicitly represents this bimodal hypothesis could be learned, at least in principle.

Finally, we also use this synthetic data problem to illustrate the significant optimization advantages that can be provided by the natural gradient updates with respect to the variational dynamics parameters. In Figure 11 we compare natural gradient updates with standard gradient updates at three different learning rates. The natural gradient algorithm not only learns much faster but also is less dependent on parameterization details: while the natural gradient update used an untuned stepsize of 0.1, the standard gradient dynamics at step sizes of both 0.1 and 0.05 resulted in some matrix parameters that are required to be positive definite to be updated to indefinite values. This particular indefiniteness error with standard gradients can be avoided by working in a different parameterization of the positive definite parameter matrices, but we used a direct parameterization to make the learning rate comparison more direct. While a step size of 0.01 yielded a stable standard gradient update, training is significantly slower than with the natural gradient algorithm, and this speed advantage of natural gradients is often greater in larger models and higher dimensions (Martens & Grosse, 2015).

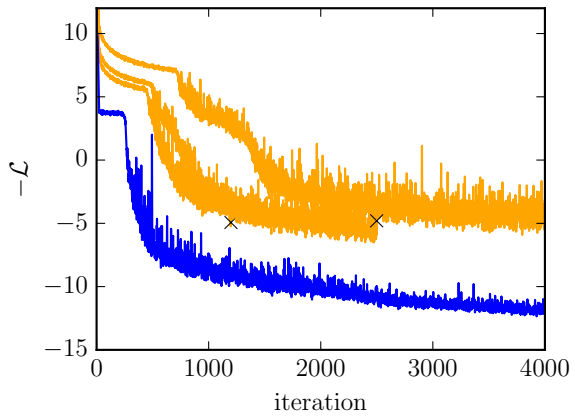


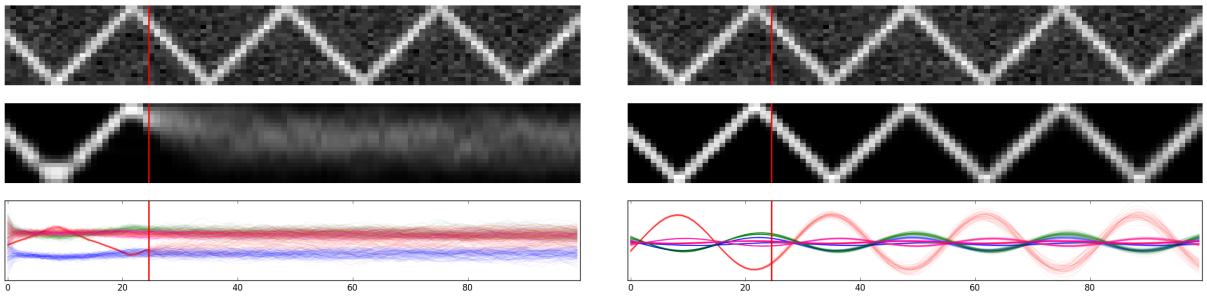
Figure 11. Comparison on the dot problem of natural gradient updates (blue) and standard gradient updates (orange). The X's indicate that the algorithm was terminated early due to indefiniteness.

6.2. Application to mouse behavioral phenotyping

As a real data experiment, we apply the SVAE to modeling depth video of freely-behaving mice. The goal of behavioral phenotyping is to identify patterns of behavior and study how those patterns change when the animal's environment, genetics, or brain function are altered. The SVAE framework is well-suited to provide new modeling tools for this task because it can learn rich nonlinear image features while learning an interpretable representation of the latent dynamics.

We use a version of the dataset from Wiltschko et al. (2015), in which a mouse is recorded from above using a depth camera, as shown in Figure 12. The mouse's position and orientation are tracked and the depth image of the mouse's body is extracted, as shown in the insets. We apply the SVAE to the task of modeling these extracted videos. We used a subset of the dataset consisting of 8 recordings, each of a distinct mouse and each 20 minutes in length at 30 frames per second, for a total of 288000 video frames.

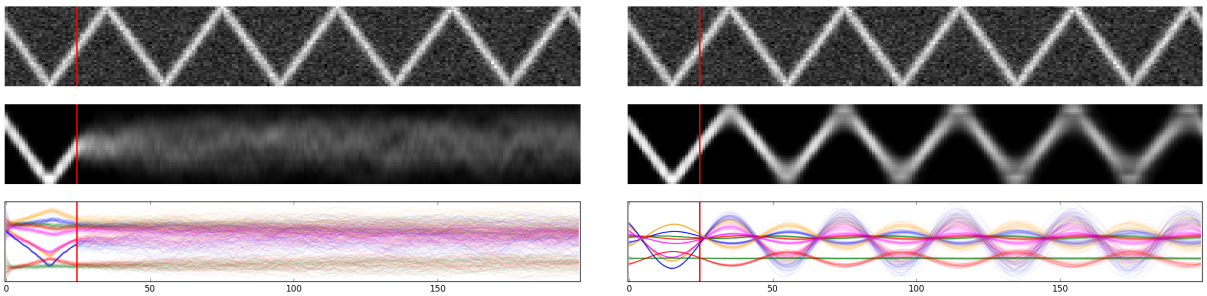
First, we show that the variational autoencoder (VAE) architecture allows us to accurately represent the manifold of depth images. We fit a standard VAE to the depth video frames, using two hidden layers of size 200 for the encoder and decoder along with a 10D latent space; we reuse this same architecture for the subsequent SVAE models. Figure 13 shows both samples of the real video frames (top panel) and images generated from the VAE by sampling from a standard Gaussian in the latent space (bottom panel), demonstrating that the VAE image model can generate very accurate synthetic mouse images. Furthermore,



(a) Predictions after 200 training epochs.

(b) Predictions after 1100 training epochs.

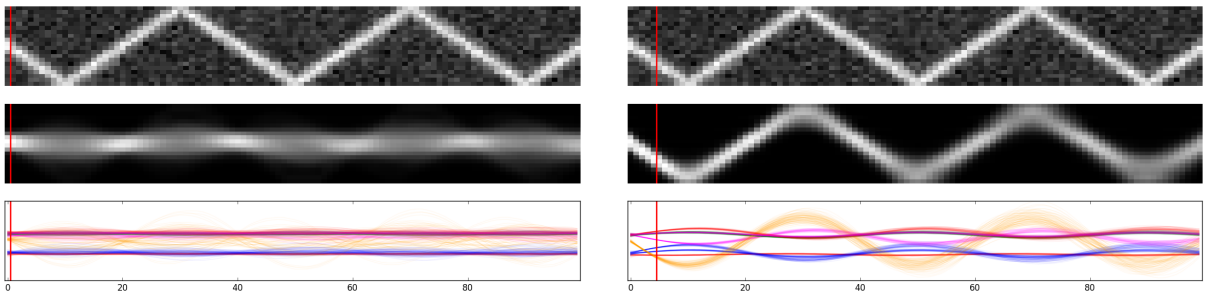
Figure 8. Predictions from an LDS SVAE fit to 1D dot image data at two different stages of training. In each subfigure, the top panel shows an example data sequence with time on the horizontal axis. The middle panel shows the noiseless LDS SVAE predictions given the data up to the vertical line, while the bottom panel shows the corresponding latent states.



(a) Predictions after 200 training epochs.

(b) Predictions after 1100 training epochs.

Figure 9. As in Figure 8 but with a higher-resolution image sequence.



(a) Prediction given one frame of data.

(b) Prediction given 5 frames of data.

Figure 10. As in Figure 8 but showing predictions given different data prefixes.

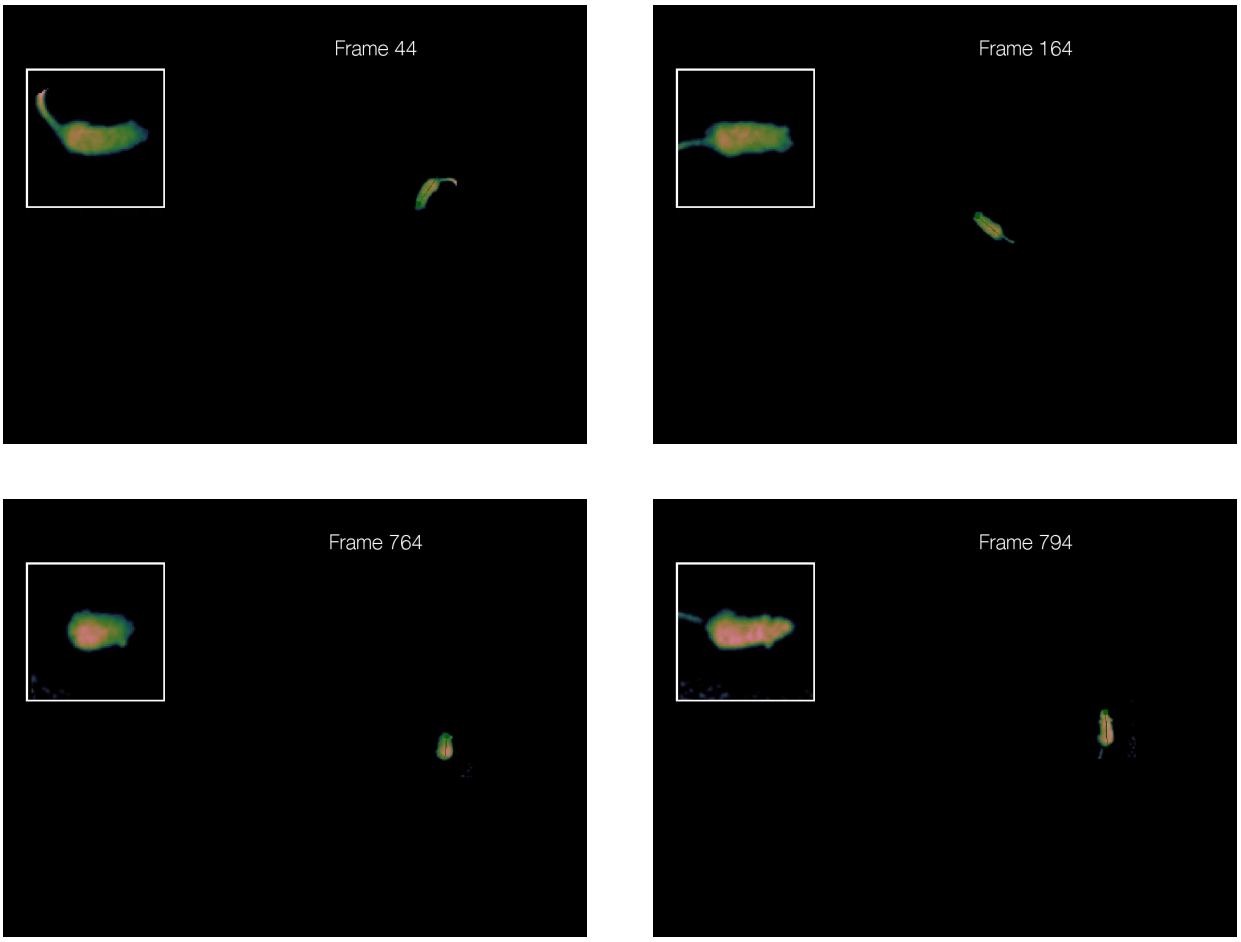


Figure 12. Mouse depth video acquisition. A freely-behaving mouse is recorded from above with a depth camera. The mouse position and orientation are tracked and the mouse body depth image is extracted, as shown in each inset. Our models are applied to these extracted image sequences after some filtering and downsampling.

to illustrate that the learned manifold naturally represents smooth variation in the mouse's pose, Figure 14 shows images generated from regular lattices on random 2D subspaces of the latent space.

While the VAE can accurately model the marginal distribution on each image, to model entire depth videos jointly we use the SVAE to learn dynamics in the latent space. To show that a Gaussian linear dynamical system (LDS) can accurately represent these latent space dynamics, we fit an LDS SVAE to the depth video sequences. Figure 15 shows predictions from this model over 1-second horizons, illustrating that the LDS SVAE retains the flexible image modeling capabilities of the VAE while additionally modeling smooth dynamics that can generate plausible video sequences.

Finally, because latent linear dynamics paired with the nonlinear image model can accurately represent the depth videos over short timescales, by extending the latent dynamics to a switching LDS (SLDS) we can discover distinct dynamical modes to represent the syllables of behavior. Figure 16 shows some of the discrete states that arise from fitting an SLDS SVAE with 30 discrete states to the depth video data. The discrete states that emerge show a natural clustering of short-timescale patterns into behavioral units. Thus the SLDS SVAE provides an interpretable, structured representation of behavioral patterns.

7. Conclusion

Structured variational autoencoders (SVAEs) provide a new set of tools for probabilistic modeling that leverage both graphical models and deep learning techniques. By building on variational autoencoders and neural network architectures, SVAEs can learn features and bottom-up inference networks automatically and represent complex high-dimensional data. By enabling the latent probabilistic models to be expressed in terms of graphical models, exponential families, and hierarchical structures, SVAEs also allow powerful modeling and algorithmic tools to be applied, including efficient natural gradient computations, graphical message passing, priors that encourage sparsity or represent domain knowledge, and discrete latent random variables. Furthermore, these structured latent representations can provide both interpretability and tractability for non-inference tasks, such as planning or control. Finally, because SVAE learning fits in the standard variational optimization framework, many recent advances in neural network optimization (Martens & Grosse, 2015) and variational inference (Burda et al., 2015) can be applied to SVAEs.

Acknowledgements

We thank Roger Grosse, Scott Linderman, Dougal Maclaurin, Andrew Miller, Oren Rippel, Yakir Reshef, Miguel Hernandez-Lobato, and the members of the Harvard Intelligent Probabilistic Systems (HIPS) Group for many helpful discussions and advice.

M.J.J. is supported by a fellowship from the Harvard/MIT Joint Grants program. A.B.W. is supported by an NSF Graduate Research Fellowship and is a Stuart H.Q. & Victoria Quan Fellow. S.R.D. is supported by fellowships from the Burroughs Wellcome Fund, the Vallee Foundation, the Khodadad Program, by grants DP2OD007109 and RO11DC011558 from the National Institutes of Health, and by the Global Brain Initiative from the Simons Foundation. R.P.A. is supported by NSF IIS-1421780.

References

Amari, Shun-Ichi. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

Amari, Shun-ichi and Nagaoka, Hiroshi. *Methods of Information Geometry*. American Mathematical Society, 2007.

Archer, Evan, Park, Il Memming, Buesing, Lars, Cunningham, John, and Paninski, Liam. Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*, 2015.

Beal, Matthew James. *Variational algorithms for approximate Bayesian inference*. University of London, 2003.

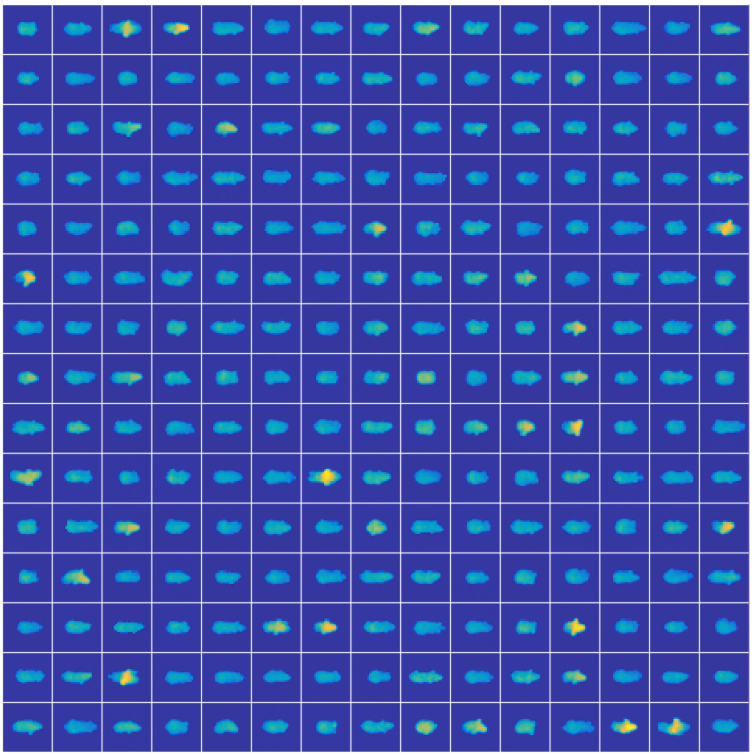
Bengio, Yoshua and Frasconi, Paolo. An input output hmm architecture. In *Advances in Neural Information Processing Systems*, pp. 427–434, 1995.

Burda, Yuri, Grosse, Roger, and Salakhutdinov, Ruslan. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

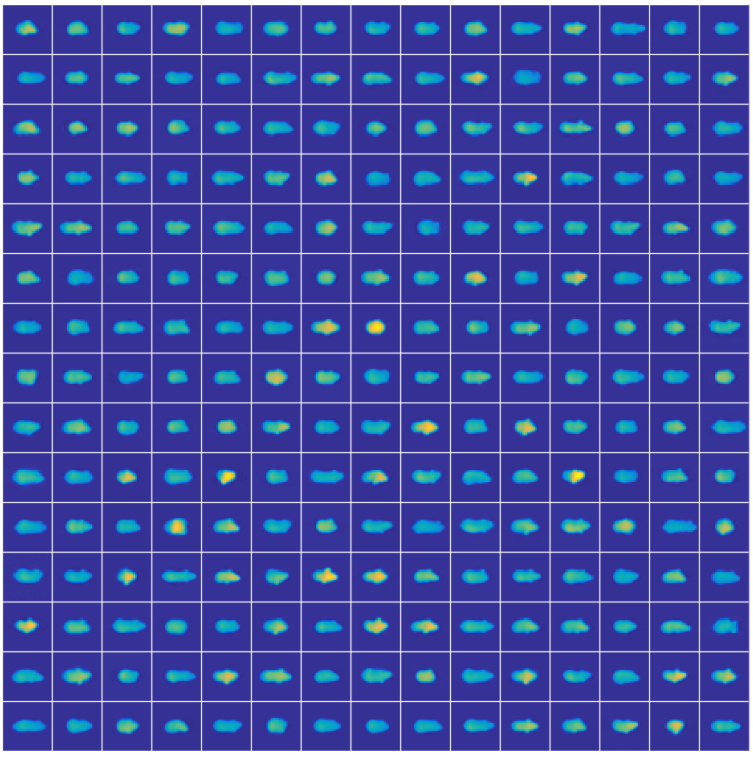
Chung, Junyoung, Kastner, Kyle, Dinh, Laurent, Goel, Kratarth, Courville, Aaron C, and Bengio, Yoshua. A recurrent latent variable model for sequential data. In *Advances in Neural information processing systems*, pp. 2962–2970, 2015.

Deng, Li. Computational models for speech production. In *Computational Models of Speech Pattern Processing*, pp. 199–213. Springer, 1999.

Deng, Li. Switching dynamic system models for speech articulation and acoustics. In *Mathematical Foundations of Speech and Language Processing*, pp. 115–133. Springer, 2004.



(a) Real frames from mouse depth video.



(b) Generated random samples from a VAE fit to mouse depth video.

Figure 13. Real data frames and generated images from a variational autoencoder (VAE) fit to mouse depth video data. In the second panel, images were generated by sampling latent coordinates from a standard 10D Gaussian.

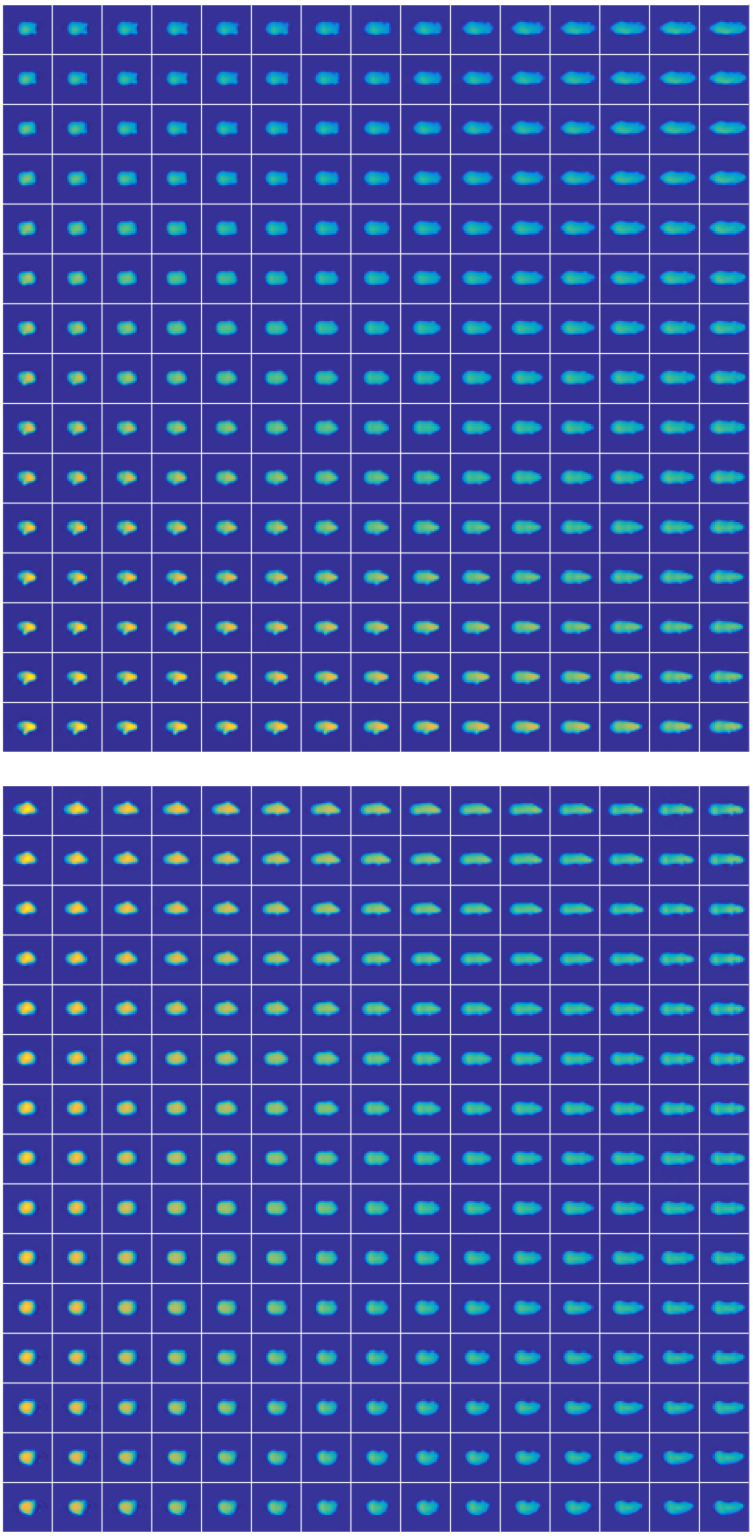


Figure 14. Generated images from variational autoencoder (VAE) fit to mouse depth video data. Each image grid was generated by choosing a random 2D subspace of the 10D latent space and sampling a regular 2D lattice of points on the subspace.

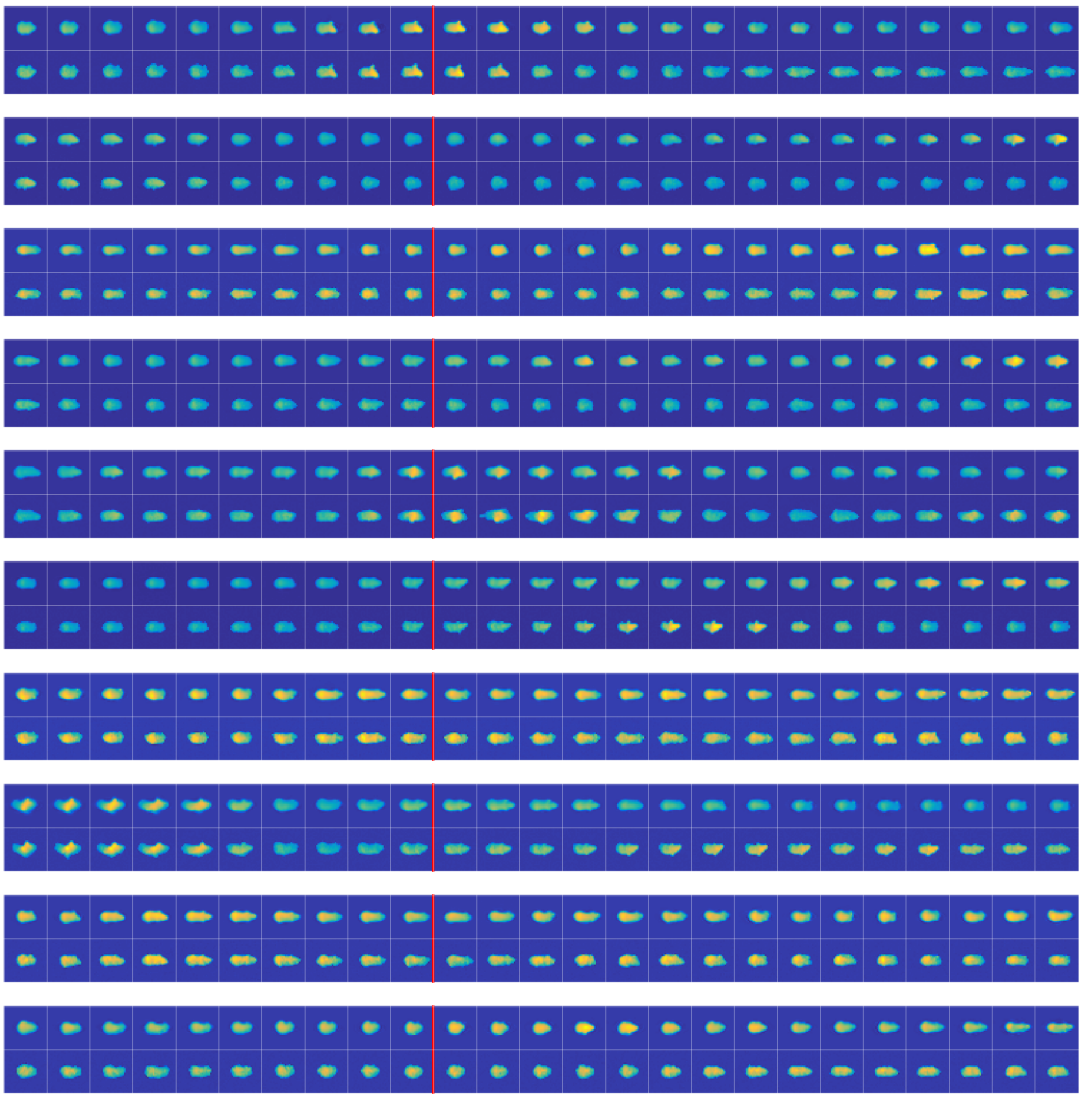
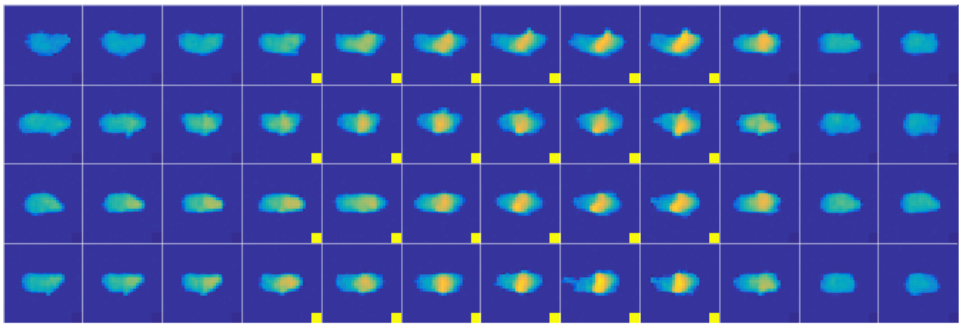
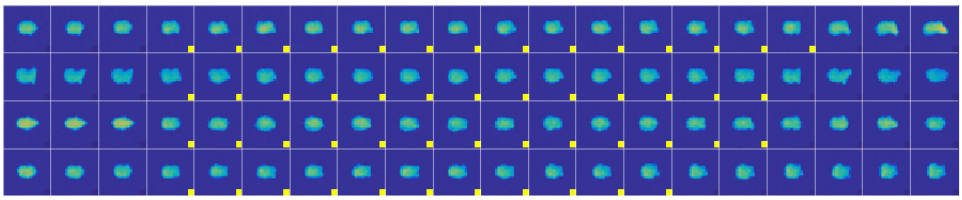


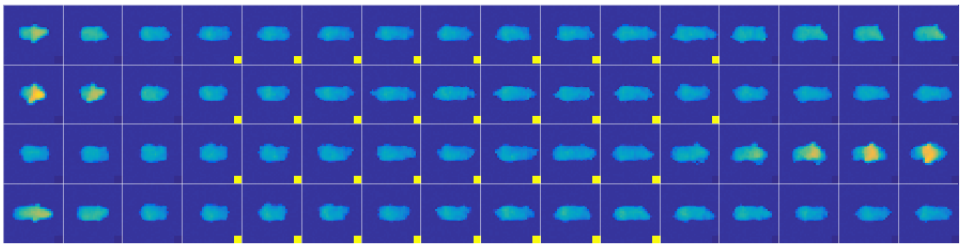
Figure 15. Examples of predictions from an LDS SVAE fit to depth video. In each panel, the top row is a sampled prediction from the LDS-SVAE and the bottom row is real data. To the left of the line, the model is conditioned on the corresponding data frames and hence generates denoised versions of the same images. To the right of the line, the model is not conditioned on the data, thus illustrating the model's predictions. The frame sequences are temporally subsampled to reduce their length, showing one of every four video frames.



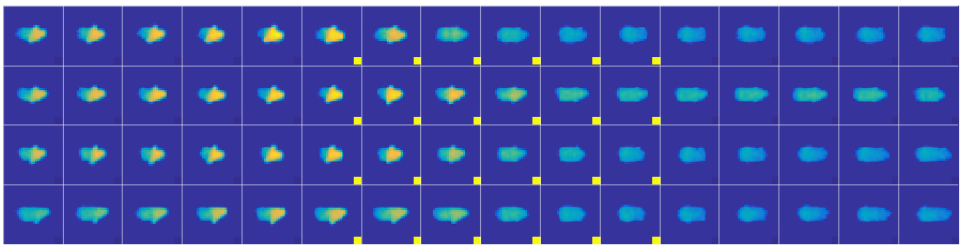
(a) Entering a rear



(b) Grooming



(c) Extension into running



(d) Fall from rear

Figure 16. Examples of behavior states inferred from depth video. For each state, four example frame sequences are shown, including frames during which the given state was most probable according to the variational distribution on the hidden state sequence. Each frame sequence is padded on both sides, with a square in the lower-right of a frame depicting that the state was active in that frame. The frame sequences are temporally subsampled to reduce their length, showing one of every four video frames. Examples were chosen to have durations close to the median duration for that state.

Foti, Nicholas, Xu, Jason, Laird, Dillon, and Fox, Emily. Stochastic variational inference for hidden markov models. In *Advances in Neural Information Processing Systems*, pp. 3599–3607, 2014.

Fox, E.B., Sudderth, E.B., Jordan, M.I., and Willsky, A.S. Bayesian nonparametric inference of switching dynamic linear models. *IEEE Transactions on Signal Processing*, 59(4), 2011.

Gelman, Andrew and Meng, Xiao-Li. Simulating normalizing constants: From importance sampling to bridge sampling to path sampling. *Statistical science*, pp. 163–185, 1998.

Gregor, Karol, Danihelka, Ivo, Graves, Alex, and Wierstra, Daan. DRAW: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

Hinton, Geoffrey, Deng, Li, Yu, Dong, Dahl, George E, Mohamed, Abdel-rahman, Jaitly, Navdeep, Senior, Andrew, Vanhoucke, Vincent, Nguyen, Patrick, Sainath, Tara N, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29 (6):82–97, 2012.

Hoffman, Matthew D., Blei, David M., Wang, Chong, and Paisley, John. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Iwata, Tomoharu, Duvenaud, David, and Ghahramani, Zoubin. Warped mixtures for nonparametric cluster shapes. In *29th Conference on Uncertainty in Artificial Intelligence*, pp. 311–319, 2013.

Johnson, Matthew J. and Willsky, Alan S. Bayesian nonparametric hidden semi-markov models. *Journal of Machine Learning Research*, 14:673–701, February 2013.

Johnson, Matthew J. and Willsky, Alan S. Stochastic variational inference for Bayesian time series models. In *International Conference on Machine Learning*, 2014.

Kingma, Diederik and Ba, Jimmy. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kingma, Diederik P. and Welling, Max. Auto-encoding variational Bayes. *International Conference on Learning Representations*, 2014.

Kleijnen, Jack PC and Rubinstein, Reuven Y. Optimization and sensitivity analysis of computer simulation models by the score function method. *European Journal of Operational Research*, 88(3):413–427, 1996.

Koller, Daphne and Friedman, Nir. *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.

Krishnan, Rahul G, Shalit, Uri, and Sontag, David. Deep Kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

Martens, James. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*, 2015.

Martens, James and Grosse, Roger. Optimizing neural networks with Kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pp. 3111–3119, 2013.

Murphy, Kevin P. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2012.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Ranganath, Rajesh, Gerrish, Sean, and Blei, David M. Black box variational inference. In *International Conference on Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Rezende, Danilo J, Mohamed, Shakir, and Wierstra, Daan. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pp. 1278–1286, 2014.

Vincent, Pascal, Larochelle, Hugo, Bengio, Yoshua, and Manzagol, Pierre-Antoine. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM, 2008.

Wainwright, Martin J. and Jordan, Michael I. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 2008.

Wiltschko, Alexander B., Johnson, Matthew J., Iurilli, Giuliano, Peterson, Ralph E., Katon, Jesse M., Pashkovski, Stan L., Abraira, Victoria E., Adams, Ryan P., and Datta, Sandeep Robert. Mapping sub-second structure in mouse behavior. *Neuron*, 88(6):1121–1135, 2015.