Automatic structured variational inference (supplementary)

A Details of the Inference Gym experiments

A.1 Inference Gym tasks

Time series models

As first application, we focus on timeseries models and SDEs. We used two models, both from the Inference Gym (Sountsov et al., 2020).

The first model (BR) is a Brownian motion without drift, governed by $x'(t) = w_x(t)$, where $w_x(t)$ is a Gaussian white noise process with scale σ_x. The value of x_t is observed with noise standard deviation equal to σ_{obs}. The data are generated with $\sigma_x = 0.1$ and $\sigma_{\text{obs}} = 0.15$. In the BRG model with global variables, both σ_x and σ_{obs} are treated as random variables with `LogNormal(loc=0, scale=2)` priors.

The second model (LZ) is a stochastic Lorenz system (nonlinear SDE):

$$x'(t) = 10(y(t) - x(t)) + w_x(t) \tag{1a}$$
$$y'(t) = x(t)(28 - z(t)) - y(t) + w_y(t) \tag{1b}$$
$$z'(t) = x(t)y(t) - (8/3)z(t) + w_z(t) \tag{1c}$$

where $w_x(t)$, $w_y(t)$ and $w_z(t)$ are Gaussian white noise processes with standard deviation $\sigma = 0.1$. The value of $x(t)$ is observed with Gaussian noise with standard deviation $\sigma_{\text{obs}} = 1.$; $y(t)$ and $z'(t$ are left unobserved. When global variables are allowed (the LZG model), σ and σ_{obs} are treated as unknown random variables with `LogNormal(loc=−1., scale=1.)` priors.

All processes were discretized with the Euler–Maruyama method ($\mathrm{d}t = 0.01$ for BR and $dt = 0.02$ for LZ) and the transition probability was approximated as Gaussian (this approximation is exact for dt tending to 0). Each model was integrated for 30 steps.

Hierarchical models

Eight Schools (Gelman et al., 2013) models the effect of coaching programs on standardized test scores, and is specified as follows:

$$\mu \sim \mathcal{N}(0, 100) \tag{2}$$
$$\log \tau \sim \log \mathcal{N}(5, 1) \tag{3}$$
$$\theta_i \sim \mathcal{N}(\mu, \tau^2) \tag{4}$$
$$y_i \sim \mathcal{N}(\theta_i, \sigma_i^2) \tag{5}$$

where $i = 1..8$ indexes the schools, μ represents the prior average treatment effect and τ controls the variance between schools. The y_i and σ_i are observed.

The Radon model (Gelman and Hill, 2007) is a Bayesian hierarchical linear regression model that predicts measurements of Radon, a carcinogenic gas, taken in houses in the United States. The hierarchical structure is

reflected in the grouping of houses by county, and the model is specified as follows:

$$\mu \sim \mathcal{N}(0, 1) \tag{6}$$
$$\tau \sim \mathcal{N}^+(0, 1) \tag{7}$$
$$\theta_i \sim \mathcal{N}(\mu, \tau^2) \tag{8}$$
$$\beta_1, \beta_2, \beta_3 \sim \mathcal{N}(0, 1) \tag{9}$$
$$\sigma \sim \mathcal{N}^+(0, 1) \tag{10}$$
$$y_j \sim \mathcal{N}(\beta_1 z_{c_j} + \beta_2 x_j + \beta_3 \bar{x}_{c_j} + \theta_{c_j}, \sigma^2) \tag{11}$$

where θ_i is the effect for county i (with prior mean μ and standard deviation τ) and the β are regression coefficients. The log Radon measurement in house j, y_j, depends on the effect θ_{c_j} for the county to which the house belongs, as well as features z_{c_j} (the log uranium measurement in county c_j), x_j (the floor of the house on which the measurement was taken), and $\bar{x}_{c_j}$ (the mean floor by county, a contextual effect). $\mathcal{N}^+(0, 1)$ indicates a Normal distribution with mean 0 and variance 1, truncated to nonnegative values.

A.2 Baselines

Mean Field ADVI

The ADVI (MF) surrogate posterior is constructed with the same procedure as the ASVI posterior, but using only the $\boldsymbol{\alpha}$ parameters, or equivalently, fixing $\boldsymbol{\lambda} = 0$. As with ASVI, therefore, the surrogate posterior for each variable is in the same distribution family as its prior. This differs slightly from Kucukelbir et al. (2017), in which surrogate posteriors are always bijectively transformed normal distributions, although we have no reason to believe that this difference is material to our experiments.

Inverse Autoregressive Flows

Inverse Autoregressive Flows (IAFs) are normalizing flows which autoregressively transform a base distribution (Kingma et al., 2016) with a masked neural network (Papamakarios et al., 2017). We build an IAF posterior by transforming a standard Normal distribution with two sequential two-layer IAFs built with `tfp.bijectors.MaskedAutoregressiveFlow`. The output of the flow is split and restructured to mirror the support of the prior distribution, and then constrained to the support of the prior (for example, by applying a sigmoid transformation to constrain values between zero and one, or a softplus to constrain values to be positive). In our experiments, we use two different-sized IAF posteriors: the "Large" IAF has 512 hidden units in each layer and the "Small" IAF has 8 hidden units in each layer.

Multivariate Normal

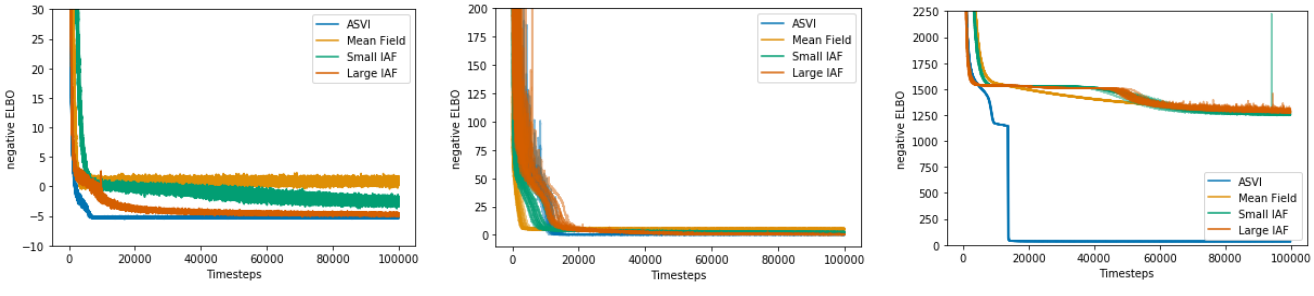
The MVN surrogate posterior is built by defining a full-covariance Multivariate Normal distribution with trainable mean and covariance, restructuring the support to the support of the prior, and constraining the samples to the prior support if necessary.

AR(1)

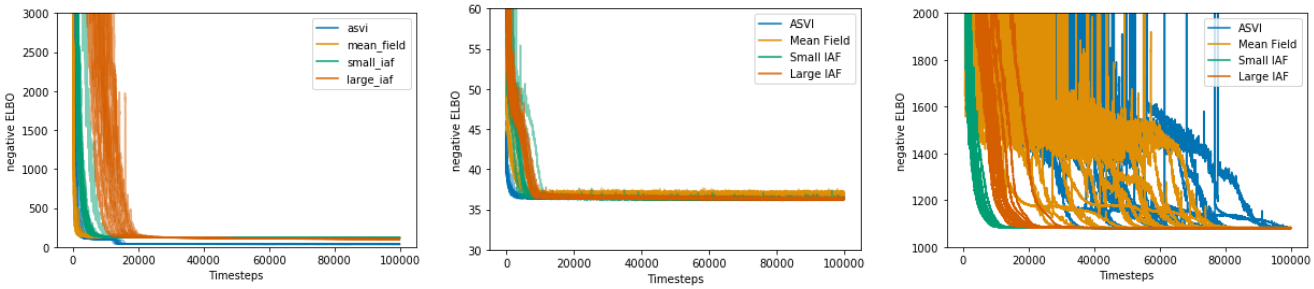
The autoregressive model surrogate learns a linear Gaussian conditional between each pair of successive model variables:

$$x_{t+1} \sim \mathcal{N}(\boldsymbol{A}_t x_t + \boldsymbol{b}_t, \boldsymbol{D}_t)$$

where each $\boldsymbol{A}_t$ and $\boldsymbol{b}_t$ parameterize a learned linear transformation, and $\boldsymbol{D}_t$ is a learned diagonal variance matrix. The linear Gaussian autoregression operates on unconstrained values, which may then be then pushed through constraining transformations as required by the model. To stabilize the optimization, we omit direct dependence on global variables, i.e., when $\boldsymbol{x}_t$ is a global variable we fix $\boldsymbol{A}_t = \mathbf{0}$ (these are generally the first few variables sampled in each model).



(a) Brownian motion (BR) (b) Brownian motion with globals (BRG) (c) Lorenz system (LZ)



(d) Lorenz system with globals (LZG) (e) Eight Schools (E) (f) Radon (R)

Figure 1: Training losses (negative ELBO values) for the ASVI, MF, Large IAF, and Small IAF baselines on the Inference Gym tasks. Each posterior was trained with the Adam optimizer for 100000 steps.

A.3 Training details

For each of the following inference tasks and posterior baselines, we fit a posterior using full-batch gradient descent on a 1-sample Monte Carlo estimate of the ELBO. We use the Adam optimizer with a learning rates selected by hyperparameter sweep: 1e-2 learning rate for ASVI, MF, and AR(1), 1e-3 for the MVN and Small IAF, and 5e-5 for the Large IAF. Each posterior was trained for 100000 iterations; in Figure 1 and Figure 2, we report the training curves for each posterior-task pair. We find that ASVI successfully converges in all tasks; in most cases, ASVI converges well before 100000 iterations, while MF, Large IAF, and Small IAF fail to converge to a good solution in a few of the tasks.

B Details of the neural SDE experiment

B.1 Models

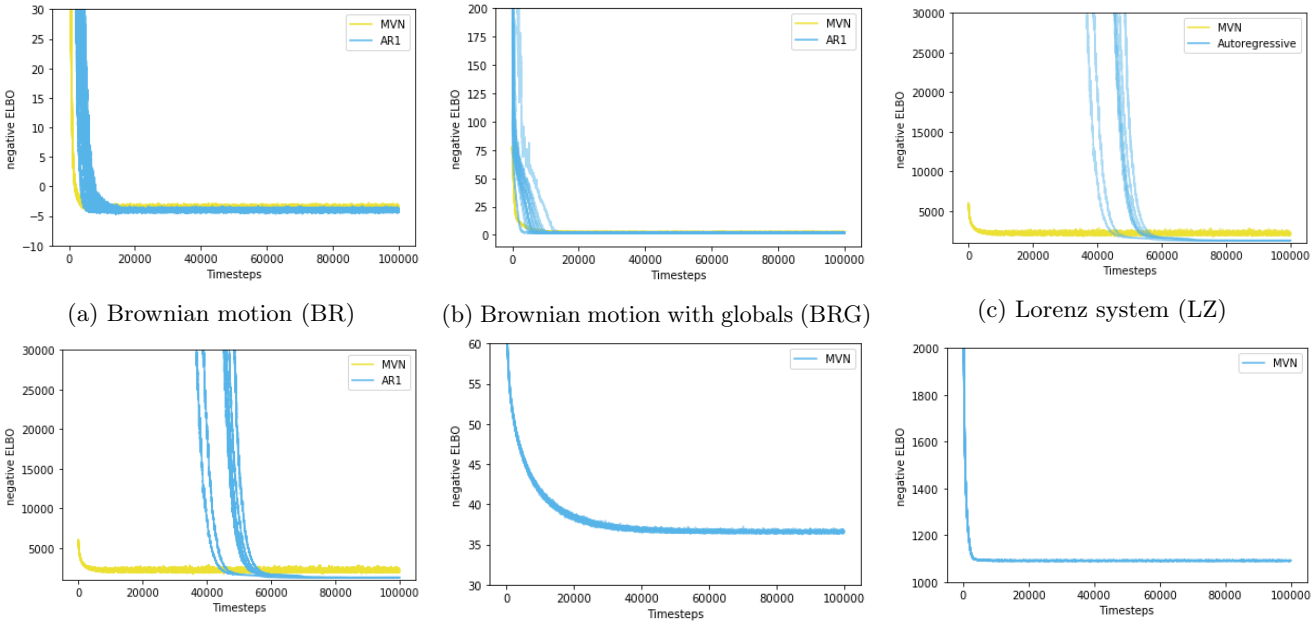
The function $\boldsymbol{F}(\boldsymbol{x})$ had the following form

$$\boldsymbol{F}(\boldsymbol{x}) = W_2 \tanh\left(\boldsymbol{x} + \tanh(W_1 \boldsymbol{x})\right) \tag{12}$$

where W_2 and W_1 were $d \times d$ matrices whose entries were sampled in each of the 5 repetitions from a centered normal with SD equal to 0.2. Those matrices encodes the forward dynamical model and they were assumed to be known during the experiment. This is a Kalman filter-like setting where the form of the forward model is known and the inference is performed in the latent units. The neural SDE was integrated using Euler–Maruyama integration with step size equal to 1 from $t = 0$ to $t = 9$. We trained the model by back-propagating though the integrator.

We used two DCGAN generators as emission models. The networks were the DCGAN implemented in PyTorch. In the CIFAR experiment, we used the following architecture:

```
ConvTranspose2d(100,64*8,4,1,0,
                bias=False),
BatchNorm2d(64*8),
ReLU(True)
```



(a) Brownian motion (BR)

(b) Brownian motion with globals (BRG)

(c) Lorenz system (LZ)

(d) Lorenz system with globals (LZG)

(e) Eight Schools (E)

(f) Radon (R)

Figure 2: Training losses (negative ELBO values) for the MVN and AR(1) baselines on the Inference Gym tasks. Each posterior was trained with the Adam optimizer for 100000 steps.

```
ConvTranspose2d(64*8, 64*4,4,2,1,
                bias=False),
BatchNorm2d(ngf*4),
ReLU(True),
ConvTranspose2d(64*4,64*2,4,2,1,
                bias=False),
BatchNorm2d(64*2),
ReLU(True),
ConvTranspose2d(64*2,64,4,2,1,
                bias=False),
BatchNorm2d(ngf),
ReLU(True),
ConvTranspose2(64,4,kernel_size=1,
               stride=1,
               padding=0,
               bias=False),
Tanh()
```

Network pretrained on CFAR was obtained from the GitHub repository: csinva/gan-pretrained-pytorch. The FashionGEN network was downloaded from the pytorch GAN zoo repository. The architectural details are given in Radford et al. (2015).

B.2 Baselines

The ADVI (MF) baseline was obtained by replacing all the conditional Gaussian distributions in the probabilistic program with Gaussian distributions with uncoupled trainable mean and standard deviation parameters. ADVI (MN) was not computationally feasible in this larger scale experiment. Therefore, we implemented a a linear Gaussian model whith conditional densities:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \sim \mathcal{N}\left(W\boldsymbol{x}_{t-1} + \boldsymbol{\alpha}_t, \boldsymbol{\sigma}_t^2\right) \ , \tag{13}$$

where the matrix W, and the vectors $\boldsymbol{\alpha}_t$ and $\boldsymbol{\sigma}_t^2$ are learnable parameters.

C Details of the autoencoder experiment

C.1 Models

Decoder 1 ($\boldsymbol{f}_1(\boldsymbol{z}_1)$)

```
hidden_size=25
Linear(latent_size1, hidden_size)
ReLU()
Linear(hidden_size, latent_size2)
```

Decoder 2 ($\boldsymbol{f}_2(\boldsymbol{z}_2)$)

```
hidden_size = 75
Linear(latent_size2, hidden_size)
ReLU()
Linear(hidden_size, latent_size3)
```

Decoder 3 ($\boldsymbol{g}_1(\boldsymbol{z}_3)$)

```
Linear(latent_size3, image_size)
```

Decoder 4 ($\boldsymbol{\alpha}(\boldsymbol{y})$)

```
Linear(latent_size3, image_size)
```

Inference network ($\boldsymbol{f}_1(\boldsymbol{z}_1)$)

```
hidden_size1=120
Linear(image_size, hidden_size)
ReLU() # For hidden units
# Latent mean output
# Latent log sd output
Linear(hidden_size1, latent_size3)
Softplus() # For standard deviation output
hidden_size2=70
Linear(latent_size3, hidden_size2)
ReLU() # For hidden units
# Latent mean output
Linear(hidden_size2, latent_size2)
# Latent log sd output
Linear(hidden_size2, latent_size2)
Softplus() # For standard deviation
hidden_size3=70
Linear(latent_size2, hidden_size3)
ReLU() # For hidden units
# Latent mean output
Linear(hidden_size3, latent_size1)
# Latent log sd output
Linear(hidden_size3, latent_size1)
Softplus() # For standard deviation
```

References

P. Sountsov, A. Radul, and contributors. Inference gym, 2020. URL `https://pypi.org/project/inference_gym`.

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, 3 edition, 2013.

A. Gelman and J. Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press, 2007.

A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474, 2017.

D. P Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, 2017.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.