A. Supplementary material

A.1. Derivation of MoG proposals for APT with MDNs

When $q_{F(x,\phi)}$ is an M-component mixture of Gaussians, $\tilde{p}(\theta)$ is an L-component mixture of Gaussians and $p(\theta)$ is Gaussian, we have

$$q_{F(x,\phi)}(\theta) = \sum_{i=1}^{M} \alpha_i \mathcal{N}_\theta\left(\mu_i, \Sigma_i\right) \tag{7}$$

$$\tilde{p}(\theta) = \sum_{k=1}^{L} \beta_k \mathcal{N}_\theta\left(\tilde{\mu}_k, \tilde{\Sigma}_k\right) \tag{8}$$

$$p(\theta|x) = \mathcal{N}_\theta\left(\mu_0, \Sigma_0\right) \tag{9}$$

$$\tilde{q}_{x,\phi}(\theta) = \frac{1}{Z_{x,\phi}} \sum_{i,k} \alpha_i \beta_k \frac{\mathcal{N}_\theta\left(\mu_i, \Sigma_i\right) \mathcal{N}_\theta\left(\tilde{\mu}_k, \tilde{\Sigma}_k\right)}{\mathcal{N}_\theta\left(\mu_0, \Sigma_0\right)} \tag{10}$$

$$= \sum_{i,k} \zeta_{ik} \mathcal{N}_\theta\left(\mu^*_{ik}, \Sigma^*_{ik}\right) \tag{11}$$

where

$$\Sigma^*_{ik} = \left(\Sigma_i^{-1} + \tilde{\Sigma}_k^{-1} - \Sigma_0^{-1}\right)^{-1} \tag{12}$$

$$\mu^*_{ik} = \Sigma^*_{ik}\left(\Sigma_i^{-1}\mu_i + \tilde{\Sigma}_k^{-1}\tilde{\mu}_k - \Sigma_0^{-1}\mu_0\right) \tag{13}$$

$$\zeta_{ik} \propto \alpha_i \beta_k \sqrt{\frac{\det(\Sigma^*_{ik})}{\det(\Sigma_i)\det(\tilde{\Sigma}_k)}}\, e^{-\frac{1}{2}\left(\mu_i^\top \Sigma_i^{-1}\mu_i + \tilde{\mu}_k^\top \tilde{\Sigma}_k^{-1}\tilde{\mu}_k - {\mu^*_{ik}}^\top {\Sigma^*_{ik}}^{-1}\mu^*_{ik}\right)} \tag{14}$$

and the proportionality symbol indicates that the weights ζ_{ik} should be normalized so that $\sum_{ik} \zeta_{ik} = 1$.

A.2. Algorithm and computational complexity for atomic APT

Algorithm 2 APT with atomic proposals

```
**Input:** simulator with (implicit) density $p(x|\theta)$, data $x_o$, prior $p(\theta)$, density family $q_\psi$, neural network $F(x, \phi)$, simulations per round $N$, number of rounds $R$, number of atoms $M$.

$\tilde{p}_1(\theta) := p(\theta)$
$c \leftarrow 0$                                                        total simulation count
**for** $r = 1$ **to** $R$ **do**
  **for** $j = 1$ **to** $N$ **do**
    $c \leftarrow c + 1$
    Sample $\theta_c \sim \tilde{p}_r(\theta)$
    Simulate $x_c \sim p(x|\theta_c)$
  **end for**
  $V_r(\Theta) := \begin{cases} \binom{c}{M}^{-1}, & \text{if } \Theta = \{\theta_{b_1}, \theta_{b_2}, \ldots, \theta_{b_M}\} \text{ and } 1 \leq b_1 < b_2 < \ldots < b_M \leq c \\ 0, & \text{otherwise} \end{cases}$      sampling without replacement
  $\phi \leftarrow \operatorname{argmin}_\phi \mathbb{E}_{\Theta \sim V_r(\Theta)} \left[ \sum_{\theta_j \in \Theta} -\log \tilde{q}_{x_j,\phi}(\theta_j) \right]$
  $\tilde{p}_{r+1}(\theta) := q_{F(x_o,\phi)}(\theta)$
**end for**
**return** $q_{F(x_o,\phi)}(\theta)$
```

To minimize $\mathbb{E}_{\Theta \sim V_i(\Theta)} \left[\sum_{\theta_j \in \Theta} -\log \tilde{q}_{x_j,\phi}(\theta_j) \right]$, we must calculate its gradient with respect to ϕ, for which we use minibatches of size M. Specifically, for each minibatch we first sample $B = \{b_1, \ldots, b_M\} \subset \{1, \ldots, c\}$ without replacement. We then calculate the gradient $\frac{d}{d\phi} \sum_{b \in B} -\log \tilde{q}_{x_b,\phi}(\theta_b)$ using (6). Note that each minibatch involves M loss evaluations—that is, M multiple-choice questions with M possible answers each.

To calculate $\tilde{q}_{x_j,\phi}$ and its gradients for a single minibatch using (6), we must evaluate $q_{F(x,\phi)}(\theta)$ on every possible pair $(\theta_b, x_{b'})$ for $b, b' \in B$. Therefore atomic APT's computational complexity is quadratic in the minibatch size M. However, we observed no difference in wallclock time per minibatch for $M = 10$ vs. $M = 100$ on an nVidia GeForce RTX 2080. Furthermore, for the Lokta-Volterra and RPS models, simulations took longer than all other calculations, for all methods.

When using an MDN as the density estimator, the MoG estimate of the posterior can be calculated once for each $x_{b'}$, and then each MoG evaluated on each θ_b. For a network with n_{layers} fully connected hidden layers of n_{hidden} units each, $d = \dim(x)$ and an n_{MoG}-component Gaussian mixture, the computational complexity of an atomic APT minibatch is

$$C_{\text{atomic MDN-APT}} = \mathcal{O}\left(M\left[dn_{\text{hidden}} + n_{\text{layers}}n_{\text{hidden}}^2 + n_{\text{hidden}}n_{\text{MoG}}\dim(\theta)^2\right] + M^2 n_{\text{MoG}}\dim(\theta)^2\right) \tag{15}$$

Note that only the final term is quadratic in M, and this term does not involve the network structure or input dimensionality. For comparison, SNPE-A/B or non-atomic MDN-based APT has the same complexity, except for being linear in M:

$$C_{\text{SNPE-A/B/non-atomic MDN-APT}} = \mathcal{O}\left(M\left[dn_{\text{hidden}} + n_{\text{layers}}n_{\text{hidden}}^2 + n_{\text{hidden}}n_{\text{MoG}}\dim(\theta)^2\right]\right) \tag{16}$$

In a MAF data and parameters are coupled, so for atomic APT every $(\theta_b, x_{b'})$ pair requires a separate feedforward pass. For n_{MADEs} conditional MADEs consisting of n_{layers} fully connected hidden layers of n_{hidden} units, the minibatch complexity is

$$C_{\text{atomic MAF-APT}} = \mathcal{O}\left(n_{\text{MADEs}}\left[Mdn_{\text{hidden}} + M^2\dim(\theta)n_{\text{layers}}n_{\text{hidden}}^2 + M^2\dim(\theta)^2 n_{\text{hidden}}\right]\right) \tag{17}$$

The first term is exempted from quadratic dependence on M since the inputs x can be multiplied by the appropriate weight matrices independently of θ. Thus, any computational tasks that scale with the input dimension d scale only linearly with M.

For MAF-based SNL the roles of the data and parameters are reversed, so the complexity (not including MCMC sampling) is

$$C_{\text{MAF-SNL}} = \mathcal{O}\left(Mn_{\text{MADEs}}\left[\dim(\theta)n_{\text{hidden}} + dn_{\text{layers}}n_{\text{hidden}}^2 + d^2 n_{\text{hidden}}\right]\right) \tag{18}$$

For very large d, we hypothesize that atomic MAF-APT could benefit from specialized network architectures that pass x through a learned, feed-forward dimensionality reduction before supplying the result as input to all MADEs.

A.3. Conditional flow normalization and truncated priors

Conditional density estimators trained to target the posterior density should respect constraints imposed on them by the prior. For uniform priors, the posterior density outside the prior support should be zero. The proposal correction for SNPE-A (Papamakarios & Murray, 2016) does not hold for tight priors that clearly truncate the posterior estimate. Also for SNPE-B, posterior leakage leads to hard-to-interpret results unless the MDNs are re-normalized.

The normalization of the conditional density model $q_{F(x,\phi)}(\theta)$ cancels out in the computation of the probabilities $\tilde{q}_{x,\phi}(\theta)$ (see eq. 6). Hence conditional density models optimized via APT with atomic proposals are automatically normalized during training. After training, the Gaussian (mixture) $q_{F(x_o,\phi)}(\theta)$ returned from MDNs can be truncated to return a valid truncated Normal posterior estimate. For MAFs we can effectively obtain such post-hoc truncation through rejection sampling, and estimate the normalization factor from the rejection rates.

We noticed that across many rounds, conditional MAFs trained with APT can leak increasingly large amounts of mass outside the prior support. We did not find this leakage to negatively influence the quality of the estimated posterior shape (evaluated over the prior support). If needed, there would be several options to reduce leakage across rounds and keep rejection rates low: We can periodically reinitialize the conditional density estimator across rounds. Alternatively, one could also train a new flow which is optimized to have the same shape on the prior's support, but minimal mass elsewhere. This normalized flow could then be used to directly evaluate posterior densities. For simple box-shaped prior supports, one can also apply a pointwise (scaled) logistic transformation to the MAF outputs to enforce prior bounds, i.e. train $q_{F(x,\phi)}(\sigma^{-1}(\theta))$.

A.4. Proof of proposition 1

Here we prove Prop. 1. Note that whenever we refer to $\tilde{p}(\theta)$, $\tilde{p}(\theta|x)$ or $\tilde{p}(x)$ these distributions are always defined based on some specific choice of Θ.

Proof of proposition 1

$$\mathbb{E}_{\Theta\sim V,\theta\sim U_\Theta,x\sim p(x|\theta)}[\tilde{\mathcal{L}}(\phi)-\tilde{\mathcal{L}}(\phi^*)] = \int_\Theta V(\Theta)\sum_{\theta\in\Theta}\tilde{p}(\theta)\int_x p(x|\theta)[\log\tilde{p}(\theta|x)-\log\tilde{q}_{x,\phi}(\theta)] \tag{19}$$

By Bayes' rule and (1), for $\theta\in\Theta$ we have $p(x|\theta)=\frac{\tilde{p}(\theta|x)\tilde{p}(x)}{\tilde{p}(\theta)}$ so

$$\begin{aligned}\mathbb{E}_{\Theta\sim V,\theta\sim U_\Theta,x\sim p(x|\theta)}[\tilde{\mathcal{L}}(\phi)-\tilde{\mathcal{L}}(\phi^*)] &= \int_\Theta V(\Theta)\int_x \tilde{p}(x)\sum_{\theta\in\Theta}\tilde{p}(\theta|x)\log\frac{\tilde{p}(\theta|x)}{\tilde{q}_{x,\phi}(\theta)} && (20)\\ &= \int_\Theta V(\Theta)\int_x \tilde{p}(x)D_{\mathrm{KL}}(\tilde{p}(\theta|x)||\tilde{q}_{x,\phi}(\theta)) && (21)\\ &= \mathbb{E}_{\Theta\sim V,\theta\sim U_\Theta,x\sim p(x|\theta)}[D_{\mathrm{KL}}(\tilde{p}(\theta|x)||\tilde{q}_{x,\phi}(\theta))] && (22)\end{aligned}$$

By Gibbs' inequality, the KL divergence is zero only when $\tilde{q}_{x,\phi}(\theta)=\tilde{p}(\theta|x)$ for all $\theta\in\Theta$, in which case by (5-6) $q_{F(x,\phi)}\propto p(\theta|x)$ for $\theta\in\Theta$ as well. Thus $D_{\mathrm{KL}}(\tilde{p}(\theta|x)||\tilde{q}_{x,\phi}(\theta))>0$ whenever $\rho(x,\Theta,\phi)\neq 1$, so if $\mathbb{E}_{\Theta\sim V,\theta\sim U_\Theta,x\sim p(x|\theta)}[\rho(x,\Theta,\phi)]$ were less than one, $\mathbb{E}_{\Theta\sim V,\theta\sim U_\Theta,x\sim p(x|\theta)}[\tilde{\mathcal{L}}(\phi)-\tilde{\mathcal{L}}(\phi^*)]$ would be greater than zero.

A.5. Experimental details

We use the same basic network architectures for all of our experiments. For mixture-density networks (SNPE-A, SNPE-B, APT), we use two fully-connected tanh layers with with 50 units each. Unless otherwise stated, we use MDNs with 8 Gaussian mixture components. In our experiments with MDNs MoG proposals were used for APT. For conditional masked autoregressive flows (SNL, APT), we use stacks of 5 MADEs each constructed using two fully-connected tanh layers with 50 units each. We train the APT MAFs with atomic proposals using $M=100$ atoms.

For the SLCP, Lotka-Volterra and M/G/1 model, we follow the experimental setup of (Papamakarios et al., 2018), including uniform priors, summary statistics, ground-truth parameters θ^* and observed data x_o.

A.5.1. Two moons model

For given parameter $\theta \in \mathbb{R}^2$, the ‘two moons’ simulator generates $x \in \mathbb{R}^2$ according to

$$a \sim U(-\frac{\pi}{2}, \frac{\pi}{2}) \tag{23}$$

$$r \sim \mathcal{N}(0.1, 0.01^2) \tag{24}$$

$$p = (r\cos(a) + 0.25,\ r\sin(a)) \tag{25}$$

$$x^\top = p + \left(-\frac{|\theta_1 + \theta_2|}{\sqrt{2}}, \frac{-\theta_1 + \theta_2}{\sqrt{2}}\right) \tag{26}$$

The intermediate variables p follow a single crescent-shaped distribution, which is subsequently shifted and rotated around the origin depending on θ. Consequently, $p(x|\theta)$ shows a single shifted crescent for fixed θ. The absolute value $|\theta_1 + \theta_2|$ gives rise to the second crescent in the posterior. We choose a uniform prior over $[-1, 1]^2$ to illustrate inference on this model. As observed data we use $x_o = (0, 0)^\top$.

On this example we fit mixture-density networks with 20 mixture components to allow expressive conditional densities.

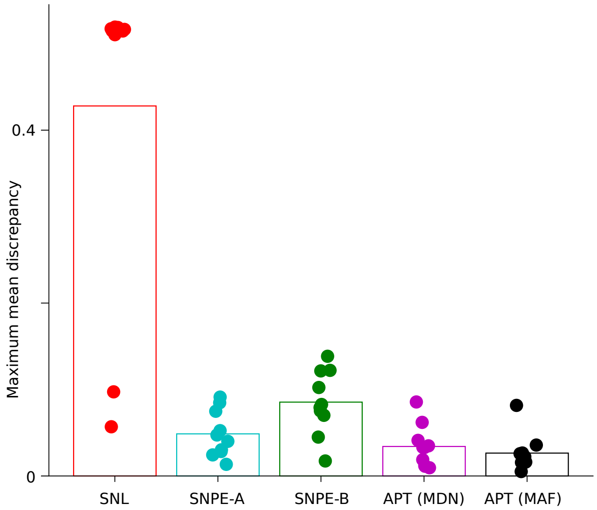

Figure 7. Two moons model. Comparison of average maximum mean discrepancies between final posterior estimate and ground-truth posteriors for different algorithms across 10 different random initializations and $x = (0, 0)$. Dots show individual runs. The MCMC chains used by SNL to obtain posterior estimates failed to sample both posterior modes in the majority of cases (cf. Fig 1 for such an example case), leading to high discrepancies.

A.5.2. SLCP model

The SLCP model has a simple simulator density $p(x|\theta) = \prod_{i=1}^4 \mathcal{N}(x_{(2i-1,2i)}|\mu(\theta), \Sigma(\theta))$, i.e. $x \in \mathbb{R}^8$ consists of four independent samples from a bivariate Gaussian parameterized by $\theta \in \mathbb{R}^5$. The conditional mean is given by $\mu(\theta) = (\theta_1, \theta_2)^\top$. The parameterization of the covariance

$$\Sigma(\theta) = \begin{bmatrix} \theta_3^2 & \theta_3\theta_4\tanh(\theta_5) \\ \theta_3\theta_4\tanh(\theta_5) & \theta_4^2 \end{bmatrix} \tag{27}$$

leads to in total four modes in $p(\theta|x)$ visible in the pairwise marginal over (θ_3, θ_4) (Fig. 2**a**). To calculate MMD’s, We sampled from the ground-truth posterior using MCMC. For the experiment with added uninformative simulator outputs, we generate noise outputs $x_{i>8} \in \mathbb{R}^m, m \in \{12, 32, 52, 92\}$ from m-dimensional mixtures of t-distributions and append them to the 8-dimensional simulator output. We use mixtures of 20 multivariate t-distributions with randomized means and covariance matrices and degree of freedom 2 to create non-trivial densities $p(x|\theta)$. We note that the actual posterior density $p(\theta|x)$ (for any noise outputs $x_{i>8}$) retains the shape of the original SLCP model due to

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} = \frac{p(x_{i>8}|x_{i\leq 8}, \theta)p(x_{i\leq 8}|\theta)p(\theta)}{p(x_{i>8}|x_{i\leq 8})p(x_{i\leq 8})} = \frac{p(x_{i>8})p(x_{i\leq 8}|\theta)p(\theta)}{p(x_{i>8})p(x_{i\leq 8})} = \frac{p(x_{i\leq 8}|\theta)p(\theta)}{p(x_{i\leq 8})} \tag{28}$$

To avoid effects of the autoregressive nature of MAF density estimators, we re-order the dimensions i of the original eight output dimensions $x_{i\leq 8}$ and our added simulator outputs $x_{i>8}$ with a fixed random permutation.

Note that in Fig. 3 the MMD for the true posterior (lower gray lines) is nonzero due to a finite number of samples being used.

A.5.3. Lotka-Volterra model

For the comparisons against previous neural conditional density models, we apply APT to infer the posterior of the Lotka-Volterra model as described in (Papamakarios et al., 2018).

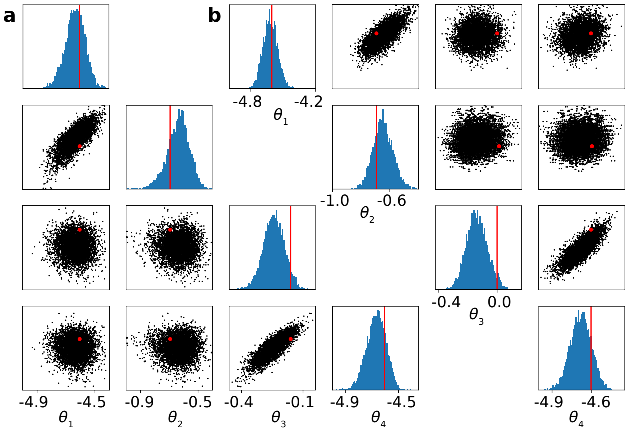

Figure 8. Lotka-Volterra model. Close-up comparison between **a** APT and **b** SNL posterior estimates.

We also infer posteriors for a Lotka-Volterra model with added observation noise. We add independent Gaussian noise $\varepsilon_{it} \sim \mathcal{N}(0, \sigma^2)$ to both populations $i = 1, 2$ and every time point $t = 0, \ldots, 150$ of the raw simulated time-series. For RNN-APT, we added an initial layer of 100 GRU units (Cho et al., 2014) to a MDN with a single Gaussian component.

A.5.4. Rock-paper-scissors model

We approximated the SPDE using a system of coupled stochastic differential equations as described in eq. (29-30) of (Reichenbach et al., 2008), and integrated this system using the Euler-Murayama method with a step size of 1 on a 100x100 grid. We calculated second spatial derivatives using simple finite differences-of-differences. We initialized the system at the unstable uniform steady state, and integrated from $t = 0$ to $t = 100$. With μ, σ and D denoting the growth rate, predation rate and diffusion constant as defined in (Reichenbach et al., 2007; 2008), θ_1, θ_2 and θ_3 were defined as their respective base 10 logarithms. We used uniform priors on each θ_j, from -1 to 1 for $1 \leq j \leq 2$ and from -6 to -5 for $j = 3$.

We used a CNN consisting of 6 convolutional layers with ReLu units, with each convolutional layer followed by a max pooling layer. The number of channels after each layer was: 8, 8, 8, 16, 32 and 32. The convolutional filter sizes were: 3, 3, 3, 3, 2 and 2. The max pooling sizes were: 1, 3, 2, 2, 2 and 1. The image sizes were 100x100, 32x32, 15x15, 6x6, 2x2 and 1x1. The 32-dimensional output of the CNN was then passed through two fully-connected tanh layers with with 50 units each. We used a single-component MDN for this problem. Overall, for this architecture the elements of ϕ (weights and biases) numbered 8768 for the CNN layers, 4200 for the fully connected layers and 612 for the final mapping onto ψ.

A.5.5. M/G/1 model

We compared MAF-APT to previous approaches on the M/G/1 queue model (as described in Papamakarios & Murray, 2016). The results (mean ± SEM) for 10 different random initializations with the identical x_o are shown in figure 9.

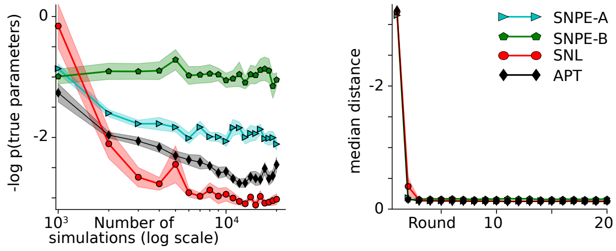

Figure 9. M/G/1 model. Averages ±1 SEM over 10 different random intializations with identical x_o ($\leq$ 10 for SNPE-A in later rounds).

A.5.6. Generalized linear model

We also compare MAF-APT against previous neural conditional density approaches on a Generalized Linear model model with a length-9 temporal input filter and a bias weight (i.e. $\theta_j, j = 1, \ldots, d$ with $d = 10$ parameters in total). We simulate 100 time bins of activity in response to white noise and summarize the output with 10 sufficient statistics x as in (Lueckmann et al., 2017). We also train the algorithms with 5 rounds of $N = 5000$ each. We note that the posteriors for this simulator are well approximated as Gaussian, and use a single Gaussian component for the MDNs for SNPE-A and -B, and MAFs consisting of two MADES for SNL and APT. We use networks with two layers of 10 tanh units for MDNs and MADES.

The results (mean ± SEM) for 10 different random initializations and x_o are shown in figure 10.

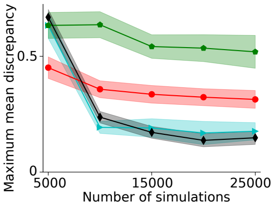

Figure 10. Generalized linear model. Averages ± 1 SEM over 10 different random intializations.