Annealed Stein variational gradient descent with local bandwidth

Zuheng(David) Xu ZUHENG.XU@STAT.UBC.CA
Student number: 24617185

1. Summary of Stein variational gradient descent

Stein variational gradient descent (SVGD) (Liu and Wang, 2016) is a particle-based variational inference method, which iteratively transforms a set of samples from a reference distribution to the target distribution. In general, SVGD only requires the gradient of the log density function of the target distribution. This fits well into the framework of Bayesian inference, which usually has intractable normalizing constant in target distribution. Comparing to traditional sampling methods such as Markov Chain Monte Carlo (MCMC), SVGD uses deterministic updates and does not experience slow convergence due to autocorrelation between samples. A big advantage is that it leverage the gradient information of the target distribution, However, it also has many shortcomings. In this section, we briefly summarize the development of SVGD and its underlying theory, and discuss its limitations at the end of this section.

1.1 KL minimization and Stein's operator

Similar to normalizing flow (Kobyzev et al., 2020), SVGD aims to learn an invertible transformation from a reference distribution to the target distribution p. Instead of composing a series of parametric functions and optimizing over all the parameters, SVGD iteratively performs a non-parametric transformation to the current distribution and decreases the KL divergence to the target.

Let X be a random variable follows a distribution with density function $q(x)$, which dominates p. Let $T : \mathbb{R}^d \to \mathbb{R}^d$ be an invertible and continuously differentiable map. In particular, we focus on the perturbed identity map:

$$T(x) := x + \epsilon\phi(x), \quad \phi \in \mathcal{H}_k^d \text{ and } \phi \text{ is smooth.}$$

Here $\mathcal{H}_k^d$ is a *Reproducing kernel Hilbert space*(RKHS), a Hilbert space induced by a positive definite kernel $\kappa(x, x') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. We denote q_T as the push forward measure of $Y := T(X)$, of which the density function can be written as

$$q_T(y) = q\left(T^{-1}(y)\right) \cdot \left|\det \nabla T^{-1}(y)\right|.$$

The development of SVGD arise from a critical connection between the KL divergence and the *Stein's operator*.

Considering the directional derivative of $\mathrm{D}_{\mathrm{KL}}\left(q_T||p\right)$ along $\phi \in \mathcal{H}_k^d$,

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\epsilon}\mathrm{D}_{\mathrm{KL}}\left(q_T||p\right)\Big|_{\epsilon=0} &= -\int q(x)\left(\nabla \log p(x)^T\phi(x) - \operatorname{tr}\nabla\phi(x)\right)\\ &= -\mathbb{E}_q\left[\nabla \log p(X)^T\phi(X) + \operatorname{tr}\nabla\phi(X)\right].\end{aligned}$$

A key observation here is that $\nabla \log p(X)^T\phi(X) + \operatorname{tr}\nabla\phi(X)$ is the *Stein operator* applied to $\phi(x)$. Then since $\phi \in \mathcal{H}_k^d$, the reproducing property of the kernel $\kappa(\cdot,\cdot)$ yields that

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon}\mathrm{D}_{\mathrm{KL}}\left(q||p\right)\Big|_{\epsilon=0} = -\left\langle\phi, \mathbb{E}_q\left[\kappa(x,\cdot)\nabla \log p(X)^T\phi(X) + \nabla_x\kappa(x,\cdot)\right]\right\rangle_{\mathcal{H}_k^d}.$$

Note that the gradient is the directional derivative along the steepest descent direction. By maximizing the right hand side over $\phi \in \mathcal{H}_k^d$, we obtain that

$$\phi^\star(\cdot) = \mathbb{E}_q\left[\kappa(x,\cdot)\nabla \log p(X)^T\phi(X) + \nabla\kappa(x,\cdot)\right], \quad \text{and} \tag{1}$$

$$\nabla_\epsilon \mathrm{D}_{\mathrm{KL}}\left(q||p\right)|_{\epsilon=0} = \|\phi^\star\|^2_{\mathcal{H}_k^d} = \mathrm{d}_\kappa(q,p). \tag{2}$$

Here $\mathrm{d}_\kappa(q,p)$ is the *kernelized Stein discrepancy* between q and p. Thus, by iteratively applying the following transformation to X_0,

$$X_0 \sim q_0, \quad X_{k+1} \leftarrow T(X_k), \quad T(X_k) := X_k + \epsilon\phi^\star(X_k), \quad k = 0, 1, \ldots. \tag{3}$$

We are able to construct a non-parametric normalizing flow that decreases the KL-divergence sequentially. As revealed in Eq. (2), the fixed point of this transformation is achieved if and only if $\mathrm{d}_\kappa(q,p) = 0$. It is worth pointing out that in general $\mathrm{d}_\kappa(q,p) = 0$ does not imply that $q = p$. In fact, this requires a proper choice of the kernel function $\kappa(\cdot,\cdot)$, which will be discussed in detail in Section 1.2.

To implement this non-parametric normalizing flow Eq. (3), Liu and Wang (2016) approximate the optimal direction Eq. (1) using Monte Carlo samples. Specifically, one may draw n samples $\{x_i^{(0)}\}_{i=1}^n$ from the reference distribution p_0 and apply the empirical version of Eq. (3) to the set of particles, i.e., for $i = 0, 1, \ldots, n$ and learning rate $\gamma_k > 0$, the numerical iteration at k-th iteration is as follows,

$$\begin{aligned}&x_i^{(k+1)} \leftarrow x_i^{(k)} + \gamma_k\hat{\phi}_k^\star\left(x_i^{(k)}\right), \quad \text{where}\\ &\hat{\phi}_k^\star(x) := \frac{1}{n}\sum_{j=1}^n\left[\kappa\left(x_j^{(k)}, x\right)\nabla \log p\left(x_j^{(k)}\right) + \nabla_{x_j^{(k)}}\kappa\left(x_j^{(k)}, x\right)\right].\end{aligned} \tag{4}$$

Ideally, this iterative procedure will move the set of particle to the target distribution, which will eventually produce samples from p. The detailed description of the standard SVGD is presented in Algorithm 1. In practice, to achieve a faster numerical convergence, Liu and Wang (2016) suggest using the AdaGrad updates with momentum—also known as RMSprop— instead of vanilla gradient descent update described in Eq. (4). Specifically, RMSprop performs the update at k-th iteration as follows: for $i = 1, 2, \ldots, n$,

$$\begin{aligned}g_{k+1} &\leftarrow \eta g_k + (1-\eta)\left(\hat{\phi}_k^\star\left(x_i^{(k)}\right)\right)^2,\\ x_i^{(k+1)} &\leftarrow x_i^{(k)} + \frac{\gamma_k}{\sqrt{g_{k+1}+\epsilon}}\hat{\phi}_k^\star\left(x_i^{(k)}\right),\end{aligned} \tag{5}$$

where $g_0 = 0 \cdot \mathbb{1}$ and all operations on vectors are elementwise. By default, the momentum factor η is set to 0.9 and the smoothing factor ϵ is set to 10^{-8}. Empirically, we do observe significantly faster convergence of the above update rule than the naive gradient descent update Eq. (4).

1.2 Choice of the kernel

Although we have formulated the construction of SVGD algorithm using some nice properties of RKHS, we have not provided any guideline of the selection of $\kappa(\cdot, \cdot)$. Note that the behaviour of SVGD is strongly dependent on the choice of the kernel. Therefore, it is helpful to understand the fundamental requirements of the kernel function.

Recall the transformation T described in Eq. (3), which is considered to be invertible and differentiable. By the Hadmard's global inverse function theorem, a sufficient condition for the invertibility of T is to have a bounded kernel function. Another concern on the kernel is related to the converging point of Eq. (3) as $k \to \infty$. As we mentioned above, the process of Eq. (3) will converge to a fixed point of the transformation T, where kernelized Stein discrepancy between the current distribution q and the target distribution p arrives at 0. Ideally, we want to have $d_\kappa(q, p) = 0$ if and only if $q = p$. A sufficient condition for this is that $\kappa(\cdot, \cdot)$ is *integrally strictly positive definition* (Liu et al., 2016), i.e.,

$$\forall 0 < \|g\|_2^2 < \infty, \quad \int_{\mathcal{X}} g(x) k\left(x, x'\right) g\left(x'\right) \mathrm{d}x \mathrm{d}x' > 0.$$

Examples of kernels satisfying these two properties are given by

$$\forall (x, x') \in \mathbb{R} \times \mathbb{R}, \quad \kappa(x, x'; h) = \exp\left(-\frac{1}{h}|x - x'|^p\right), \quad p \in (0, 2].$$

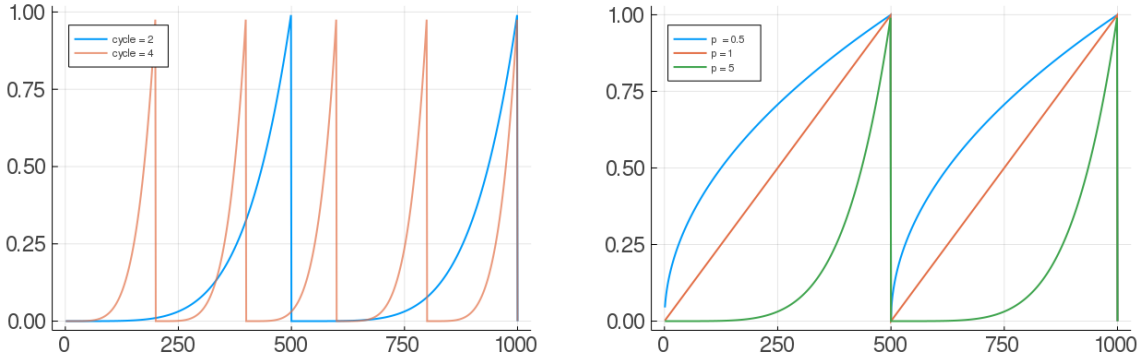
When $p = 2$, this yields the RBFkernel. Here $h > 0$ denotes the bandwidth of the kernel, which is a critical hyperparameter.

Different values of the kernel bandwidth can significantly influence the quality of SVGD; users need to hand tune the bandwidth to optimize the performance of SVGD. When using RBF kernel, Liu and Wang (2016) suggests picking the bandwidth based on the median of the pairwise Euclidean distances between the current particles, i.e., $h = \mathrm{med}^2 / \log n$, where med is the median of the Euclidean distances. This procedure will update the bandwidth automatically at each iteration and works generally well. However, we will show in Section 2 when the median trick can be problematic.

1.3 Limitations of SVGD

A key limitation of SVGD is its computational burden. Note that the deterministic updates described in Eq. (4) require the evaluation of the kernel matrix and its gradient, which costs $\mathcal{O}(n^2)$. This makes SVGD less practical compared to other particle sampling methods such as sequential Monte Carlo sampler—which costs $\mathcal{O}(n)$—if we aim to obtain a large number of samples from the target distribution.

The computation problem also occurs when calculating the gradient $\nabla logp(x)$ for all particles; this is particularly expensive in a large dataset setting. Considering the target distribution as the posterior distribution of a Bayesian model, i.e., $p(\theta) \propto \pi_0(\theta) \prod_{i=1}^m p(x_i \mid \theta)$



(a) (b)

Figure 1: Annealing schedules. Fig. 1a shows two annealing schedules with 2 cycles and 4 cycles respectively. Fig. 1b shows annealing schedules across $p = 0.5, 1, 5$.

where π_0 denotes the prior distribution of θ, computing $\nabla \log p(\theta)$ exactly is not practical as m is large. Stochastic gradient estimates using subsampled data points can be applied in this case to make SVGD scalable.

In the following sections, we focus on another common issue of SVGD when the target distribution $p(x)$ has multiple modes: the mixing problem, where the particles collapse at a particular mode of the target distribution and fail to distribute the particles to other high density regions. This is also known as the *mode collapse* phenomenon (D'Angelo and Fortuin, 2021; Zhuo et al., 2018) and appears often when the modes are distant from each other. The problem of mode collapsing tends to be more severe in high-dimensional settings, but can also be observed on a simple 2D Gaussian mixture target. Variants of SVGD are developed to resolve this issue (or at least alleviate the problem). For example, D'Angelo and Fortuin (2021) adopts the tempering method from the MCMC literature (Zhang et al., 2019) that includes an annealing schedule in the SVGD updates; Zhuo et al. (2018) leverages the latent structure of the statistical model and converts the inference problem into a lower-dimensional space.

In the next section, we provide a clear illustration of the mixing issue on two synthetic examples and introduce the annealed Stein variational gradient descent (A-SVGD), which tends to reduce the mixing issue. Further, we identify a central problem regarding the kernel bandwidth, which inspires our idea in Section 3.

2. Annealed Stein variational gradient descent

In this section, we provide a brief introduction of the annealed SVGD (A-SVGD), which is developed to address the mixing issue (D'Angelo and Fortuin, 2021), and use a few synthetic examples to identify a key limitation of the current work. The design of the synthetic example aims to understand the influence of the kernel bandwidth on the behaviour of SVGD or A-SVGD and help us to build the intuition of picking a proper value. All the experiments performed in this report use the RMSprop update rule Eq. (5). We run both

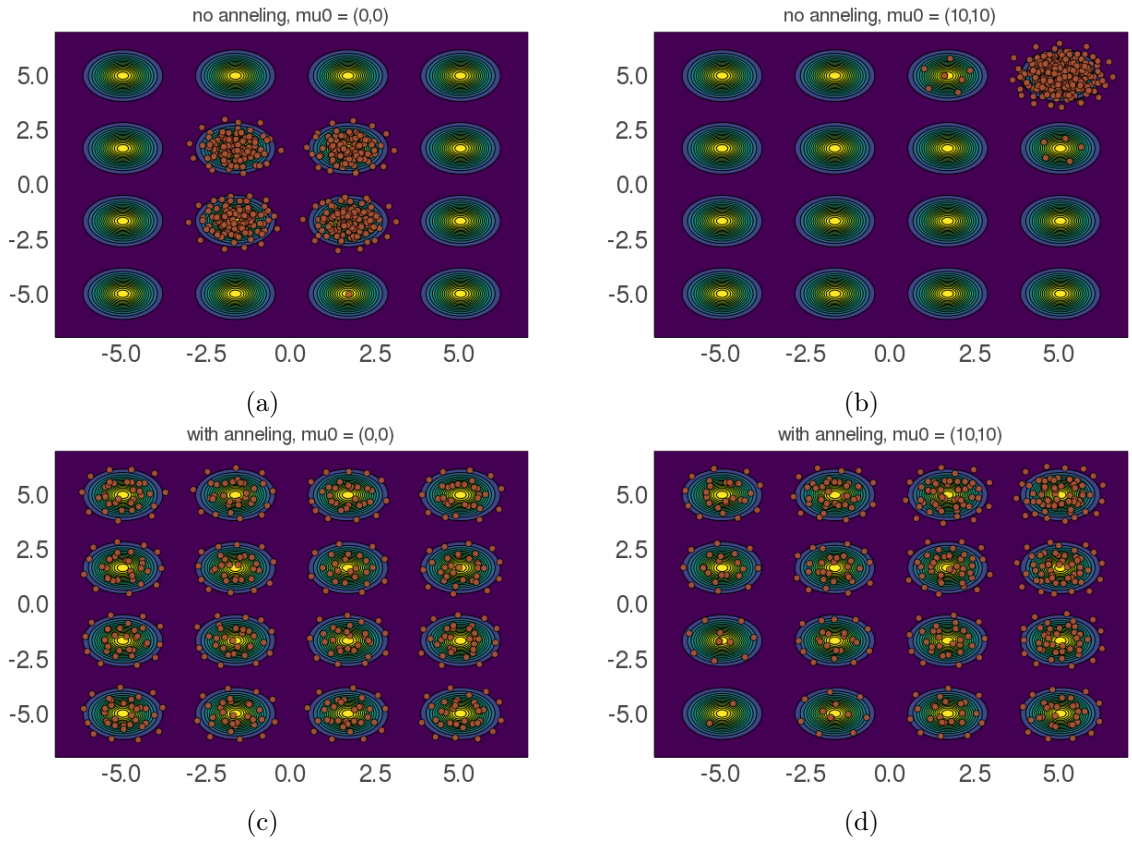


Figure 2: Comparison of SVGD (Figs. 2a and 2b) and A-SVGD (Figs. 2c and 2d) on the synthetic Gaussian mixture with 16 components with $\sigma = 0.5$ across two set of initial particles.

SVGD and A-SVGD with standard RBF kernel for 1000 iterations under a constant learning rate $\gamma_k = 0.1$.

We start with the development of A-SVGD. Considering the updates Eq. (4), the optimal descent direction $\hat{\phi}^\star(x)$ for the particle x includes two parts: the driving force that moves the particle to a high density region of $p(x)$, and a repulsion that prevents the particles collapsing together. The specific derivation is as follows,

$$\hat{\phi}_k^\star(x) = \frac{1}{n}\sum_{j=1}^{n}\left[\underbrace{\kappa\left(x_j^{(k)}, x\right)\nabla \log p\left(x_j^{(k)}\right)}_{\text{driving force}} + \underbrace{\nabla_{x_j^{(k)}}\kappa\left(x_j^{(k)}, x\right)}_{\text{repulsion}}\right].$$

When mode collapse occurs, particles produced by SVGD are trapped at some region and fail to explore the full support of $p(x)$. This often happens when the driving force dominates the repulsion. An intuitive solution is to boost the repulsion or scale down the driving force to enable a wider spread of the particles. D'Angelo and Fortuin (2021) follows this intuition

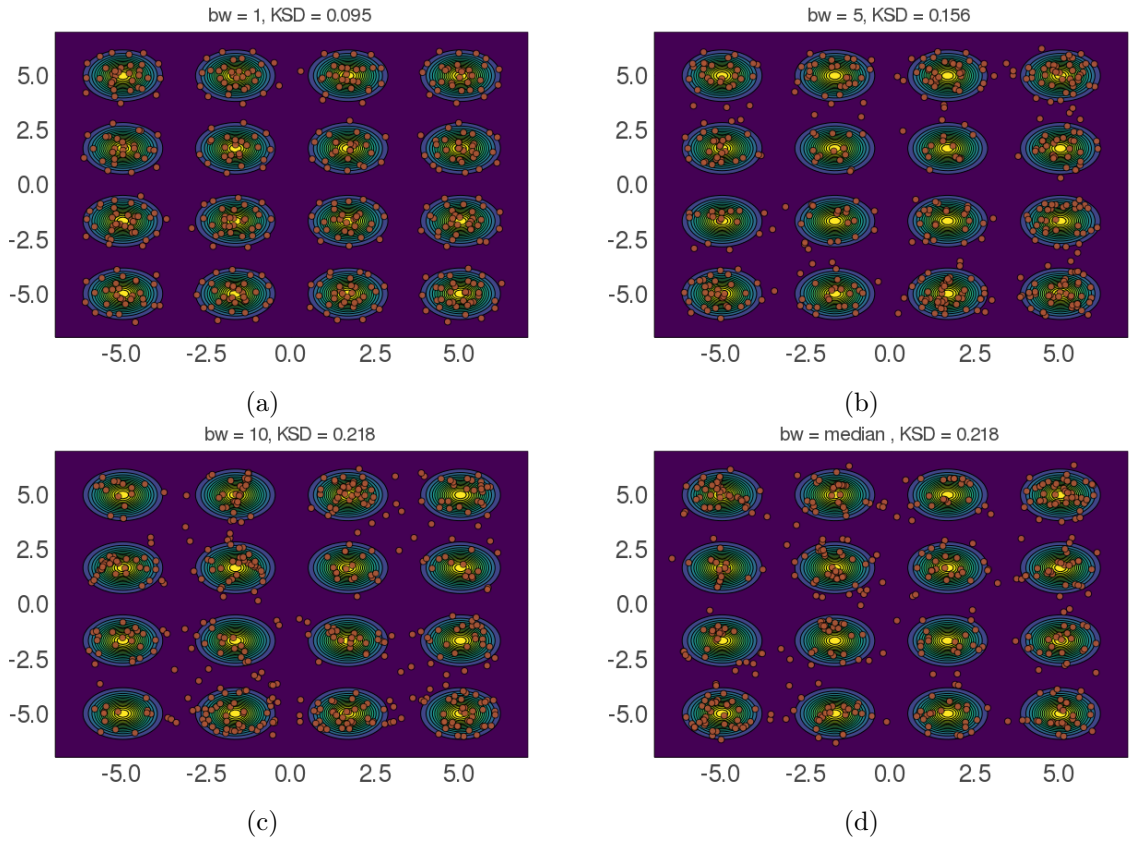


Figure 3: Comparison of A-SVGD on the synthetic Gaussian mixture with 16 components across different bandwidth.

and modify the descent direction by introducing an annealing term $\alpha(k) \in [0, 1]$:

$$\hat{\phi}_{\alpha_k}(x) = \frac{1}{n} \sum_{j=1}^{n} \left[\alpha(k) \kappa \left(x_j^{(k)}, x \right) \nabla \log p \left(x_j^{(k)} \right) + \nabla_{x_j^{(k)}} \kappa \left(x_j^{(k)}, x \right) \right]. \tag{6}$$

In particular, the authors suggest using a cyclical annealing schedule, i.e.,

$$\alpha(k) = \left(\frac{\mathrm{mod}(k, K/C)}{K/C} \right)^p,$$

where K is the total number of iterations, C is the number of cycles, and $p > 0$ is an exponent factor that controls the speed of the transition between two phases. As demonstrated in Fig. 1, smaller p corresponds to a shorter period of the exploratory phase. In all our experiments using A-SVGD, we set $p = 1$ and $C = 2$.

Now we compare the performance of A-SVGD and SVGD on a synthetic Gaussian mixture (Fig. 2). The target distribution is a 2D mixture of 16 Gaussian component with means equally distributed on a 4×4 grid with identical covariance $0.25I$. We consider two different initialization particles: Figs. 2a and 2c are initialized with 500 i.i.d. samples from $\mathcal{N}(0, 0.25I)$;

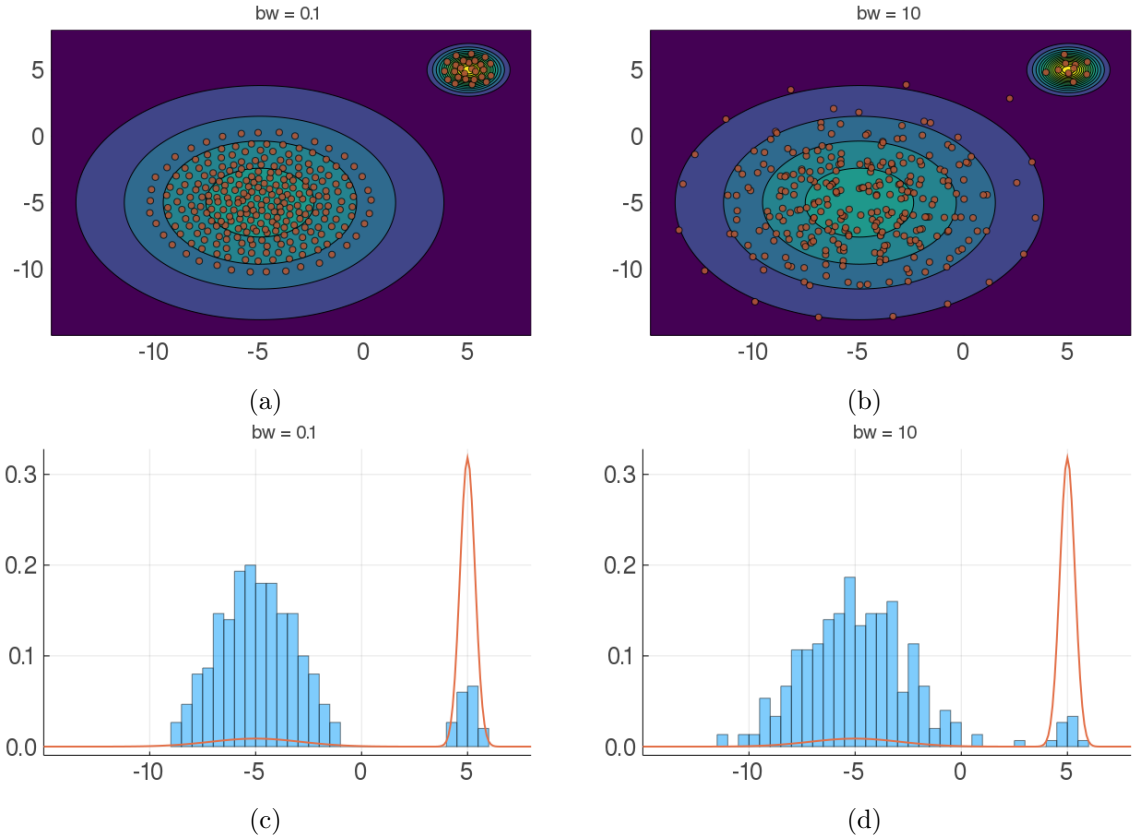
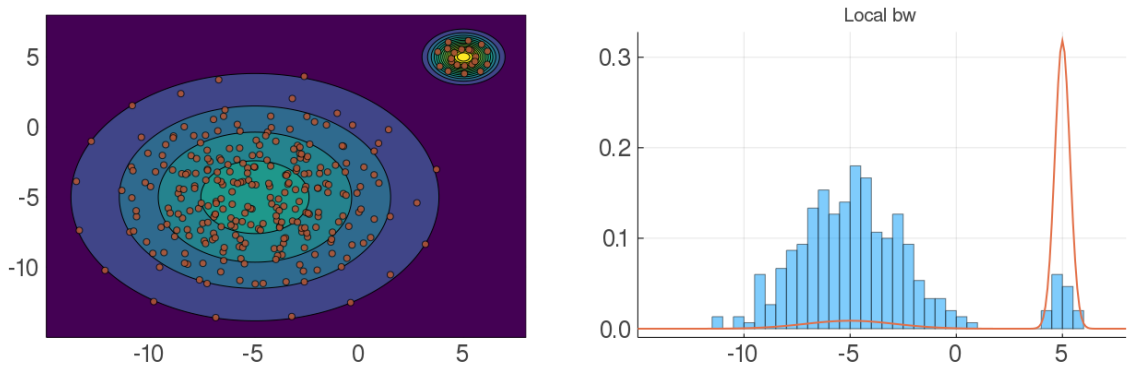


Figure 4: Comparison of A-SVGD on a 2-component Gaussian mixture with different bandwidth. Figs. 4a and 4b show scatter plots of particles at the last iteration. Figs. 4c and 4d show histograms of the particles by projecting the locations to the line $y = x$, where the orange curve denotes the density of Gaussian mixture along the slice $y = x$.

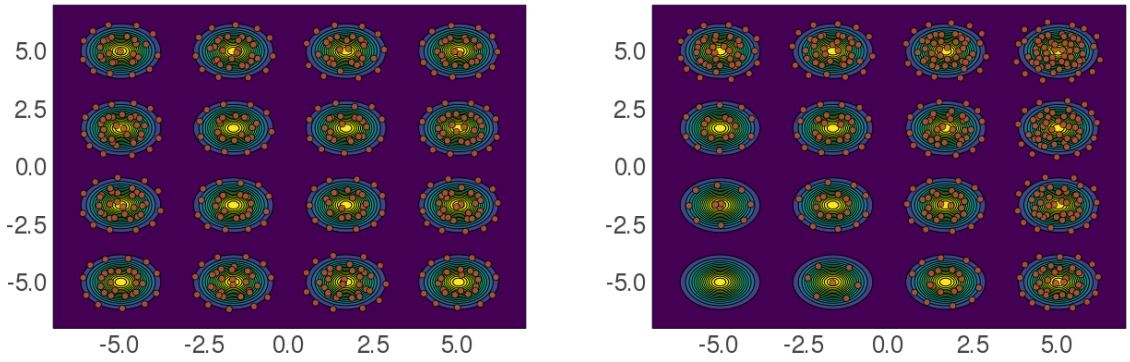
Figs. 2b and 2d are initialized with 500 i.i.d. samples from $\mathcal{N}(10, 0.25I)$. In this example, we use RBF kernel with $h = 0.5$. Note that in both initialization schemes, A-SVGD is able to explore all the modes while all the particles produced by SVGD collapse at the components close by the initialization. Comparing Figs. 2c and 2d, one may notice that the performance of A-SVGD still depend on the location of the initial particles.

Although the A-SVGD manages to obtain a better mixing, its performance relies heavily on a good choice of kernel bandwidth. Fig. 3 present the results of A-SVGD on the same target across 4 different kernel bandwidth: $h = 1, 5, 10$ and the median heuristic described in Section 1.2. The initial particles are 500 i.i.d. samples from $\mathcal{N}(0, 0.25I)$. We find that as h increases (not even a large amount), the convergence of A-SVGD gets ruined. Moreover, the median trick suggested by Liu and Wang (2016) completely fails. For quantitative characterization of the sample qualities, we compute the kernelized Stein discrepancy,which



(a) (b)

Figure 5: A-SVGD with adaptive local bandwidth on a 2-component Gaussian mixture.



(a) $\{x_i^{(0)}\}_{i=1}^{500} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0\mathbb{1}, 0.25I)$ (b) $\{x_i^{(0)}\}_{i=1}^{500} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(10\mathbb{1}, 0.25I)$

Figure 6: A-SVGD with adaptive local bandwidth on a 4×4 Gaussian mixture.

shows a decreasing sample quality as h increases. The key reason is that as the annealing schedule enables the points to traverse over different modes in the exploratory phase, the particles need to quickly converge to the high-density region around them in the converging phase. As we mentioned previously, the driving force will push the samples along the $\nabla \log p(x)$. However, the driving force is a weighted summation of $\nabla \log p(x_i), i = 1, 2, \ldots, n$ and weights are controlled by the kernel. Adopting a large bandwidth tends to include misleading information from the particles that converge to other modes.

Even though the previous example prefers a smaller bandwidth, it does not mean that we should start with a small h. In this experiment, we create a pathological yet simple example such that neither small nor large bandwidth is satisfying. Considering a two-component Gaussian mixture:

$$\frac{1}{2}\mathcal{N}\left(-5\mathbb{1}, 9I\right) + \frac{1}{2}\mathcal{N}\left(5\mathbb{1}, 0.25I\right).$$

Fig. 4 display the results of A-SVGD with 300 initial particles sampled from $\mathcal{N}(0\mathbb{1}, 25I)$ using different kernel bandwidth $h = 0.1, 10$. With small h particles located in the wide

Gaussian component are overly contracted; with large h, almost all particles converges to the wide mode. This is essentially due to the fact that the curvature of the local basin of two gaussian components is quite different. When using a wide kernel bandwidth, the driving force of those particles around the narrow Gaussian component is significantly influenced by gradient information $\log p(x)$ of the particles that are far away. As most particles are trapped in the wider Gaussian component, the driving force tends to point to the wide mode. Also, a large bandwidth leads to a stronger repulsion, which encourages the spread of particles. On the contrary, with small bandwidth—as shown in Figs. 4a and 4c—although the peaky Gaussian absorbs more particles, due to a weak repulsion introduced by the small bandwidth value, particles at the wider Gaussian component are not spread enough, resulting in an underestimation of the variance. This hints at an important fact that using a universal kernel bandwidth for all particles is not ideal, and we should use different kernel bandwidths when updating each of the particles depending on its local basin of $\nabla \log p(x)$. In the next section, we propose an adaptive local bandwidth selection scheme that requires no input from the user.

3. A-SVGD with adaptive local bandwidth

With the intuition obtained in the last section, we suggest using different bandwidth values for each particle and updates the bandwidth adaptively across the iteration. The goal of this section is to design an adaptive procedure that learns a proper kernel bandwidth for each particle automatically. As illustrated in Fig. 3 and Fig. 4, a careful choice of kernel bandwidth is necessary when the target distribution has multiple modes and one might consider using different bandwidth values at a different location. Intuitively, a proper choice of the bandwidth should reflect the local geometry of $\log p(x)$—the curvature of $\log p(x)$—which includes information of the Hessian $\nabla^2 \log p(x)$.

We consider the anisotropical Gaussian kernel with covariance Σ,

$$\kappa(x, x'; \Sigma) = \exp(-(x - x')^T \Sigma^{-1} (x - x')).$$

For computational reason, we propose to set Σ to the absolute value of $\operatorname{diag}\left(\nabla^2 \log p(x)\right)$.

Based on the empirical findings in Section 2, we find narrower mode requires smaller bandwidth while wider mode prefers a larger kernel bandwidth. Therefore, when updating x_i, an ideal choice of h would be $1/\sigma_1(\nabla^2 \log p(x_i))$, where $\sigma_1(M)$ denotes the maximal singular value of matrix M. However, solving $\sigma_1(\nabla^2 \log p(x_i))$ is generally too expensive. We then use the maximal diagonal element of $\nabla^2 \log p(x_i)$ as approximation. The complete procedure of A-SVGD using adaptive local bandwidth is described in Algorithm 2.

With other settings fixed, we apply Algorithm 2 to the previous two synthetic examples. As demonstrated in Fig. 5, by using the local bandwidth, the samples in the wide component correctly characterize its variance and there is a certain amount of samples concentrate at the peak mode. This combines the advantages of both small h and large h. And for the 16-component Gaussian example, since all Gaussian component shares the same covariance, we do not expect A-SVGD with adaptive local bandwidth could outperform the previous results significantly. As shown in Fig. 6, the result is quite similar to the A-SVGD using hand-tuned bandwidth; but importantly this requires no user input in terms of kernel bandwidth.

It is worth pointing out that leveraging the Hessian of $\log p(x)$ introduces additional computation complexity. Evaluating the Hessian for all particles costs $\mathcal{O}(d^2N)$, denoting a quadratic growth as d increases. This makes SVGD less practical in a high dimensional setting. In that case, we might consider using Gauss-Newton approximation to the Hessian matrix or other methods that approximate the second-order derivative using first-order information. Also, when n is large, evaluating $\nabla^2 \log p(x)$ for all the particles is impossible. But employing stochastic estimates of the exact Hessian matrix is a reasonable approach.

References

D'Angelo, F. and Fortuin, V. (2021). Annealed Stein variational gradient descent. *arXiv preprint arXiv:2101.09815*.

Kobyzev, I., Prince, S., and Brubaker, M. (2020). Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

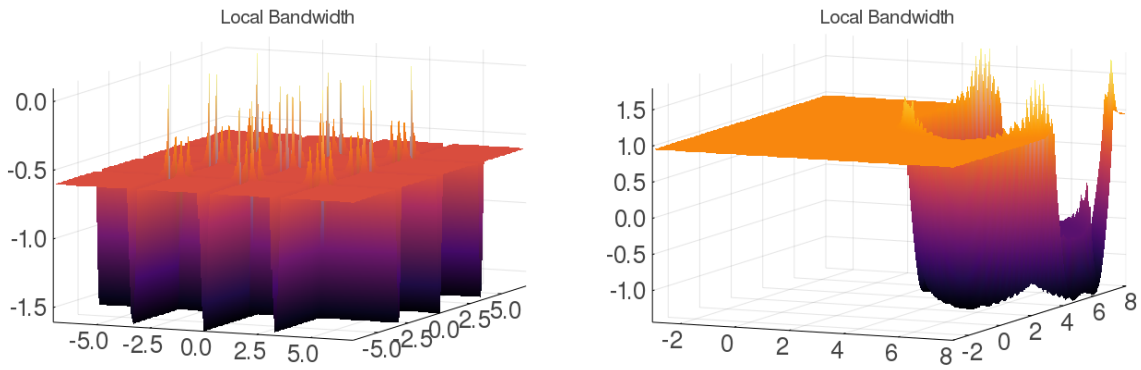
Liu, Q., Lee, J., and Jordan, M. (2016). A kernelized stein discrepancy for goodness-of-fit tests. In *International Conference on Machine Learning*, pages 276–284.

Liu, Q. and Wang, D. (2016). Stein variational gradient descent as gradient flow. In *Advances in Neural Information Processing Systems*, pages 2370–2378.

Zhang, R., Li, C., Zhang, J., Chen, C., and Wilson, A. (2019). Cyclical stochastic gradient mcmc for bayesian deep learning. *arXiv preprint arXiv:1902.03932*.

Zhuo, J., Liu, C., Shi, J., Zhu, J., Chen, N., and Zhang, B. (2018). Message passing Stein variational gradient descent. In *International Conference on Machine Learning*, pages 6018–6027.

Appendix A. Local bandwidth of two synthetic example



(a) (b)

Figure 7: Local bandwidth value for 4×4 Gaussian grid (Fig. 7a) and 2-component Gaussian mixture (Fig. 7b).

Appendix B. Algorithms

Algorithm 1 SVGD

procedure SVGD($\{x_i^{(0)}\}_{i=1}^n$, $\nabla \log p(x)$, $\kappa(\cdot,\cdot)$,γ_k)
  for $k = 0, 1, \ldots, K-1$ **do**
    for $i = 1, 2, \ldots, n$ **do**
      $x_i^{(k+1)} \leftarrow x_i^{(k)} + \gamma_k \hat{\phi}_k^\star \left(x_i^{(k)}\right)$ where $\hat{\phi}_k^\star(\cdot)$ is defined in Eq. (4)
    end for
  end for
  return $\left\{x_i^{(K)}\right\}_{i=1}^n$
end procedure

Algorithm 2 A-SVGD with local bandwidth

procedure SVGD($\{x_i^{(0)}\}_{i=1}^n$, $\nabla \log p(x)$, $\nabla^2 \log p(x)$,γ_k, $\alpha(t)$)
  for $k = 0, 1, \ldots, K-1$ **do**
    for $i = 1, 2, \ldots, n$ **do**
      $\Sigma_i \leftarrow 1/(\max\left\{\text{diag}\,\nabla^2 \log p\left(x_i^{(k)}\right)\right\}$
      $\kappa(x, x'; \Sigma_i) \leftarrow \exp(-(x-x')^T \Sigma_i^{-1}(x-x'))$
      $x_i^{(k+1)} \leftarrow x_i^{(k)} + \gamma_k \hat{\phi}_{\alpha_k}\left(x_i^{(k)}\right)$ where $\hat{\phi}_{\alpha_k}(\cdot)$ is defined in Eq. (6)
    end for
  end for
  return $\left\{x_i^{(K)}\right\}_{i=1}^n$
end procedure