Stochastic Modeling - Final Project
Stein Variational Methods for Bayesian Inference

Felipe Suárez Colmenares

1 Introduction

High-dimensional sampling of multimodal un-normalized distributions appears in many real world modeling applications where unknown continuous parameters are treated as stochastic. Yet, reasoning about such distributions faces many computational and statistical difficulties such as estimating $\mathbb{E}_{x\sim\pi}[h(x)]$ with low variance and bias, producing samples from π, and even evaluating π itself. A clear example where this mathematical challenge arises is in Bayesian inverse problems. In this project we are going to explore the applicability of a recent tool that leverages [on] the idea of particle transport and exploits the reproducibility properties of Reproducing Kernel Hilbert Spaces (RKHS) called Stein Variational Descent Methods [Liu and Wang, 2016, Detommaso et al., 2018].

In inverse problems we are interested in inverting for parameters for which we do not have access but we can measure or control some variables that interact with it, for instance through a state equation given by a partial differential equation. Some landmark examples are given by the diffusion equation that finds application in heat transfer and climate modeling; or the Helmholtz equation useful in radar and seismic imaging.

Consider a PDE $L(\theta)$ parametrized by a field θ acting on a function u with a forcing f. The function u satisfies the equation $L(\theta)u = f$ and let's call the solution of such equation as a function of the parameter $u = \mathcal{F}(\theta)$. If our measurements come from a Gaussian noise with power σ_m^2, we can model our data to be

$$d = \mathcal{F}(\theta) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma_m^2 I)$$

We would also like to include our prior knowledge about the true solution θ, maybe coming from physical intuition or previous measured data. This amounts to prescribing a prior density to θ. For the sake of exposition, let us assume that it is Gaussian centered at the origin with variance σ_p^2:

$$\theta \sim \mathcal{N}(0, \sigma_p^2 I).$$

Incorporating the knowledge of our measurements translates the problem to reasoning about the posterior $\theta|D$, which we know has an un-normalized density proportional to the likelihood times the prior:

$$\pi(\theta) := f_{\theta|D}(\theta|D) \;\alpha\; f_{D|\theta}(D|\theta) \cdot f_\theta(\theta).$$

Hence we are in a situation where

- The evaluation of π requires to compute $\mathcal{F}(\theta)$ at least once.
- π is an un-normalized density of the posterior.
- θ might be high dimensional.

The main idea of Stein Variational method is to transport a fresh sample of a cheaper distribution q toward our target π. Particle transportation can be done exactly in one dimension via the inverse function method, when the target is continuous and we have access to the inverse cumulative function. This method intuitively maps a uniform set of particles onto π-distributed particles. Furthermore, Stein Method uses an optimization approach in order to construct the best map possible from q to π.

Variational approaches in importance sampling are useful for iteratively obtaining better biasing distributions through the minimization of the KL divergence between distributions in a parametrized family Θ and π. These approaches are often difficult when π is not well captured by Θ or when the parametrization in Θ is difficult to optimize over. The Stein variational method exploits the structure of a RKHS to overcome the problems in the choice of Θ, making both the expressivity and the optimization more tractable.

2 Stein Variational Method

The goal is to obtain a transport map T from a distribution q to a possibly un-normalized distribution π in $\mathbb{R}^d$. We consider the transport map to be a small perturbation of the identity: $T = I + Q$ where Q is a function in a Hilbert space $\mathcal{H}$. To achieve distributional equivalence, we minimize the KL divergence between the transported (push forward) distribution T_*q and the target π. Let's call this objective function J:

$$J_q[Q] = KL((I + Q)_*q||\pi).$$

The three key lemmas [Detommaso et al., 2018, Liu and Wang, 2016] that are derived from the RKHS properties, namely the reproducing property[1], permit us say that:

1. The gradient of J_q vanishes when $q\ \alpha\ \pi$.
2. The maximal gradient of J_q over a unit ball can be computed exactly and efficiently.
3. An approximation of a Newton-scaled gradient can be computed exactly and efficiently.

We will make use of these lemmas to construct first and second order optimization schemes to reach a minimum of J_q. For a probability distribution p on $\mathbb{R}^d$, consider the Stein Differential operator $\mathbb{D}_p(f) := \frac{1}{p}\nabla(pf)$, and the expectation functional $\mathbb{E}_p(f) := \mathbb{E}_{x\sim p}[f(x)]$. Note that $\mathbb{D}_\pi$ is homogeneous in π, that is $\mathbb{D}_\pi = \mathbb{D}_{Z\cdot\pi}$ for any constant Z. Hence we don't require π to be normalized.

[1] Any function in $f \in \mathcal{H}$ satisfies $f(x) = \langle f(\cdot), k(x, \cdot)\rangle$.

2.1 Gradient of J

Since our optimization is carried over a function space of perturbations, we need to consider the Frecht derivative. The Frecht derivative is defined similarly as in the finite-dimensional case: the derivative at R along the Q direction is

$$DJ[R](Q) := \frac{\delta}{\delta Q} J[R] = \lim_{h \to 0} \frac{J[R + hQ] - J[R]}{h}.$$

It is not hard to see that $DJ_q[0](Q) = -tr\mathbb{E}_q\mathbb{D}_\pi(Q)$, see proof of proposition 1 in [Detommaso et al., 2018]. The functional $Q \mapsto tr\mathbb{E}_q\mathbb{D}_\pi(Q)$ is called the Stein Discrepancy and it offers a way to gauge how q and π differ from each other. Indeed, when $q = \pi$ (or any multiple of π), we have that $tr\mathbb{E}_q D_\pi(Q)$ for any smooth function Q. Hence, the gradient of J_q vanishes at the point where we reach our objective.

2.2 Maximal gradient

In doing a first order optimization iteration, we need to look for the optimal descent direction, which is dictated by the gradient. The optimal direction is that which maximizes the minus gradient over a unit ball: $\arg\max_Q tr\mathbb{E}_q\mathbb{D}_\pi(Q)$. Fortunately, since $Q \in \langle k \rangle^d$, we can expand any function Q in terms of k and obtain a exact form [Liu and Wang, 2016]:

$$\arg\max_{Q \in \langle k \rangle^d} \; tr\mathbb{E}_q\mathbb{D}_\pi(Q) = \mathbb{E}_q\mathbb{D}_\pi(k(x, \cdot)).$$

Notice that all of the quantities in $\mathbb{E}_q\mathbb{D}_\pi(k(x, \cdot))$ can be approximated efficiently. $\mathbb{E}_q$ can be computed from the samples given that q is easy to sample from, and $\mathbb{D}_\pi$ only needs access to the gradient of $\log \pi$:

$$\mathbb{D}_\pi(k(x, \cdot)) = \nabla_x \log \pi(x) \otimes k(x, \cdot) + \nabla_x k(x, \cdot).$$

Thus we can easily estimate the optimal direction to perform descent in terms of k.

2.3 Newton correction of Gradient

We can go even further and calculate the Hessian of the J to try to find the best second-order descent direction S through the Newton algorithm:

$$D^2 J[0](R, S) = -DJ[0](R), \qquad \forall S \in \langle k \rangle^d.$$

The expression for $D^2 J[0](R, S)$ was shown [Detommaso et al., 2018] to be equal to

$$D^2 J_q[0](R, S) = -tr\mathbb{E}_q(\nabla^2 \log \pi R S^T - \nabla R \nabla S).$$

Solving the full Newton-iteration has some problems: the direction is a descent direction only when $\nabla^2 \log \pi$ is log-concave, and the solution cannot be derived in terms of k in general.

Instead, we make two assumptions to overcome these difficulties:

1. Use Gauss-Newton approximation of the Hessian. That is the outer product of the Jacobian: JJ^T.

2. Solve the system at the sample points of q, decoupling the interaction between the Gradient and the Hessian at different sample points. (Solving $H(S, R) = J(R)$ locally)

2.4 Algorithmic Implementations

We will now focus in their algorithmic implementations.

Algorithm 1 Stein Variational Gradient Descent

procedure SVGD(Forward Model M, Kernel K, Samples $S = \{x_i\}_i$, Target π)
  for $i = 1, \dots, N$ **do**
    $\text{Grad}(z) \leftarrow \frac{1}{|S|} \sum_{x \in S} \nabla_x \log \pi(x) \cdot k(x, z) + \nabla_x k(x, z)$
    $S \leftarrow \{x + \eta \text{Grad}(x)\}_{x \in S}$
  end for
end procedure

Algorithm 2 Stein Variational Gauss-Newton-Galerkin

procedure SVN(Forward Model M, Kernel K, Samples $S = \{x_i\}_i$, Target π)
  for $i = 1, \dots, N$ **do**
    $\text{Grad}(z) \leftarrow \frac{1}{|S|} \sum_{x \in S} \nabla_x \log \pi(x) \cdot k(x, z) + \nabla_x k(x, z)$
    $\text{Hess}(z) \leftarrow \frac{1}{|S|} \sum_{x \in S} \nabla_x \log \pi^{\otimes 2} k(x, z)^2 + \nabla_x k(x, z)^{\otimes 2}$
    $S \leftarrow \{x + \eta \text{Hess}(x)^{-1} \text{Grad}(x)\}_{x \in S}$
  end for
end procedure

Notice that the algorithm for the Newton iteration is not using the full Hessian information $\nabla^2 \log \pi$ but instead its Gauss-Newton alternate $\nabla \log \pi^{\otimes 2}$. Also, notice that this expression was derived [Detommaso et al., 2018] after expanding in the subspace spanned by the functions $\{k(x, \cdot)\}_{x \in S}$ and not the full RKHS, hence the name Galerkin.

3 Results

In this section we are going to evaluate the performance of algorithms 1 and 2 in several model cases and using alternate descent methods. We will also record average time over 10 replicas of each experiment.

Our setup includes 3 different models in $\mathbb{R}^2$ and 2 different Kernel choices for both algorithms. Furthermore, for SVGD (algorithm 1) we are going to use a Constant Rate, Momentum, AdaGrad and RMSprop algorithm. For SVN (algorithm 2) we will use constant rate 0.1, 0.4, 0.8.

The (forward) **models** we will consider are the following:

- **Gaussian**

$$\theta = \theta_1 - 2\theta_2, \qquad \mu_p = (2, -3)^T, \sigma_p^2 = 3, \qquad D = 4, \sigma_p^2 = 4.$$

- **Anti-Banana**

$$\theta = \log((1-\theta_2)^2 + 100(\theta_2 - \theta_1^2)^2), \qquad \mu_p = (0,2)^T, \sigma_p^2 = 8, \qquad D = 13, \sigma_p^2 = 8.$$

- **Star**

$$\theta = \log((1-\theta_2^2)^2 + 10(\theta_1^2 - \theta_2^2)^2), \qquad \mu_p = (0,0)^T, \sigma_p^2 = 80, \qquad D = 5, \sigma_p^2 = 3.$$

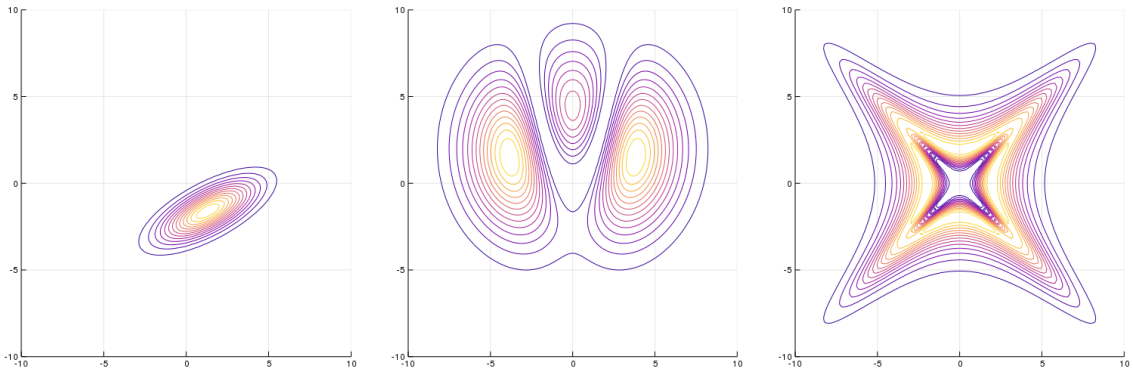

Figure 1: Contour plots of the target distribution π for different models.

The **Kernel** we will consider are an isotropic Radial Basis Function (RBF) kernel and a Covariance-scaled kernel.

- **Istropic RBF**

$$k(x,y) = \exp\left(-\frac{\|x-y\|^2}{2h^2}\right).$$

- **Covariance-scaled**

$$k(x,y) = \exp\left(-(x-y)^T J(x-y)\right).$$

The covariance used is the estimated Hessian of the log posterior from the samples. Finally, the **optimization algorithms** that we will consider [Ruder, 2018] are the following.

- **Constant Rate**(η)

$$p^+ \leftarrow \eta \nabla_\theta J, \qquad \theta^+ \leftarrow \theta - p^+.$$

- **Momentum**(η, γ)

$$p^+ \leftarrow \gamma p + \eta \nabla_\theta J, \qquad \theta^+ \leftarrow \theta - p^+$$

- **AdaGrad**(η, ϵ)

$$p_i^+ \leftarrow p_i + (\nabla_i J)^2 \qquad \theta_i^+ \leftarrow \theta_i - \frac{\eta}{\sqrt{p_i^+ + \epsilon}} \nabla_i J$$

- **RMSprop**(η, γ, ϵ)

$$p_i^+ \leftarrow \gamma p_i + (1-\gamma)(\nabla_i J)^2 \qquad \theta_i^+ \leftarrow \theta_i - \frac{\eta}{\sqrt{p_i^+ + \epsilon}} \nabla_i J$$

Table 1: 10-averaged time comparison

			Gaussian	Anti-Banana	Star
SVGD	Const.	Isotropic	19.0170	22.8149	23.9481
		Scaled	22.4338	23.6127	22.7800
	Momen.	Isotropic	18.9879	20.3262	20.7268
		Scaled	20.7791	21.1997	20.3466
	AdaG.	Isotropic	21.2317	19.3193	22.7492
		Scaled	20.5222	16.8899	23.5848
	RMS.	Isotropic	19.9172	21.3909	20.2917
		Scaled	20.9754	16.6124	21.1148
SVN	0.1	Isotropic	26.8094	25.6020	35.7627
		Scaled	28.1014	33.1475	36.5644
	0.4	Isotropic	27.2385	35.2637	35.0233
		Scaled	27.0940	40.0487	35.3709
	0.8	Isotropic	26.2704	29.0698	33.9078
		Scaled	25.4599	36.1746	35.8823

In the next table, we show the time performance of evaluation of these algorithms for a set of 512 standard Gaussian random points and 1024 iterations. The code was based on the one used in [Detommaso et al., 2018] but implemented in Julia `https://github.mit.edu/felipesc/SteinVariate`.

The only difference in the algorithms comes from solving n $d \times d$ extra linear systems in algorithm 2 over algorithm 1. For low dimensions this cost just adds larger constants to the $O(n^2)$ operations of evaluation the pairwise kernel and its derivative. Similarly, the extra time taken for calculating the scaled kernel is dominated by the $d \times d$ systems solves. Observe that the time elapsed also fluctuates with the type of model, but in these cases, the difference is not too great. The only major difference is the expression for the forward and Jacobian, which can indeed differ greatly when the Forward models are PDEs of very different nature.

Lastly, we show the results of the transported particles. We plot the maximum and average *norm of the gradients* as a function of iteration. We recommend using the interactive HTML that permits a flexible exploration of the results where SVGD and SVN are contrasted synchronously `https://github.mit.edu/pages/felipesc/felipe.github.io/SV.html`.

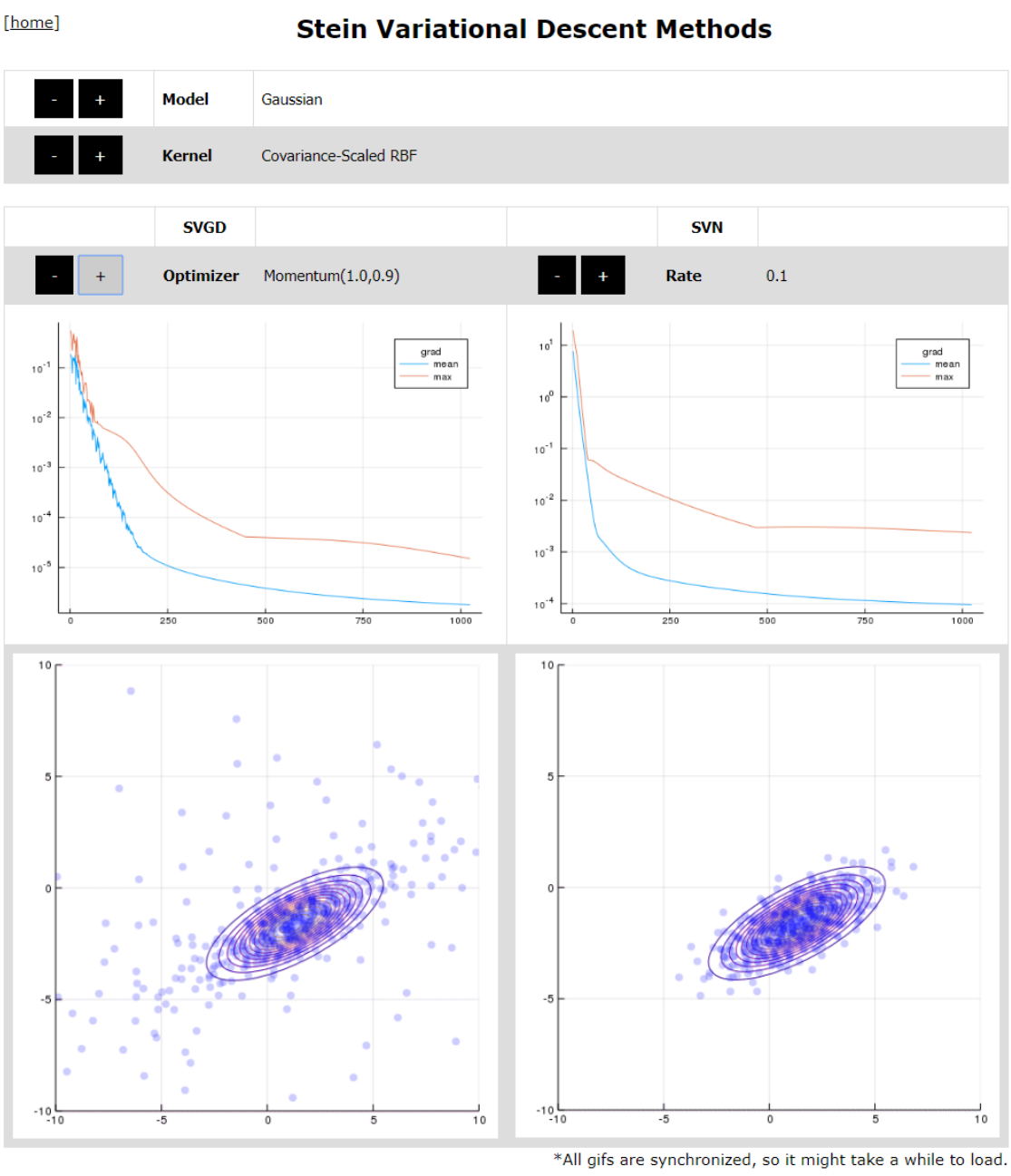

Figure 2: Screenshot of web-based result evaluation.

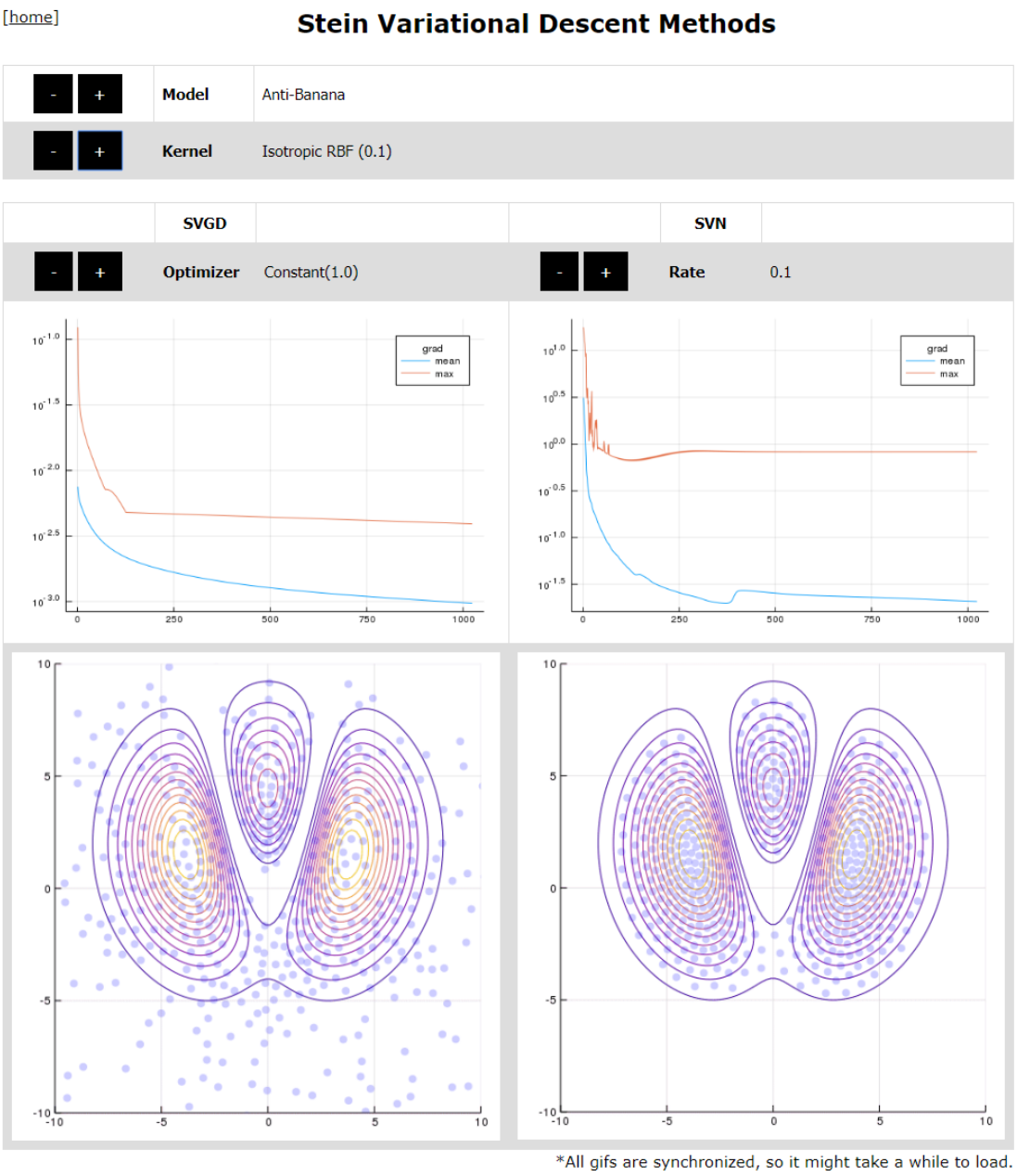

Figure 3: Screenshot of web-based result evaluation.

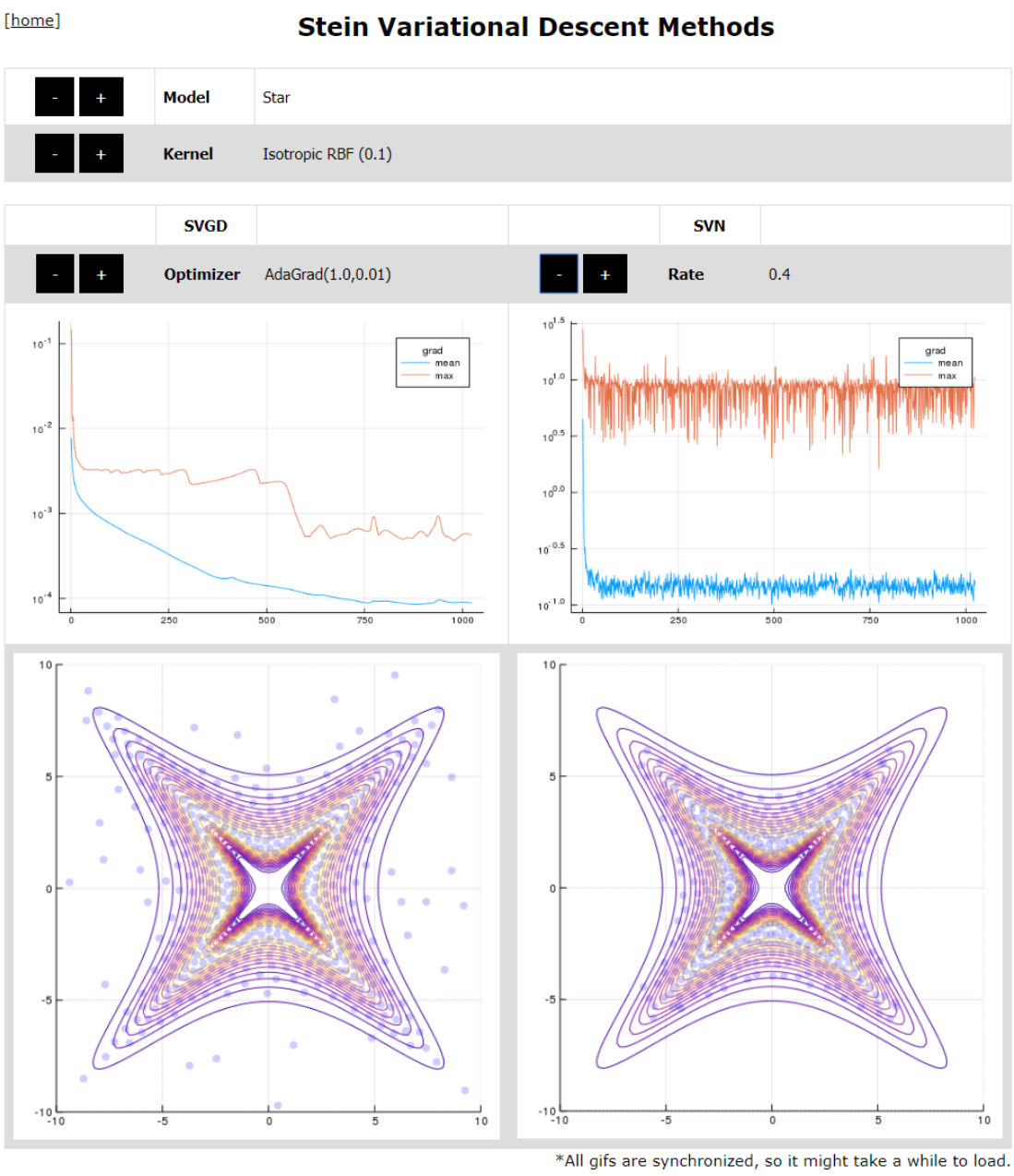

Figure 4: Screenshot of web-based result evaluation.

The results obtained from the implementation of these algorithms are manifold. The first clear remark is that SVN converges to its minimum twice as fast when it actually converges, but it does it to worse local minimum. In addition, the problem seems to suffer from many curses in non-convex optimization. Namely, extreme sensitivity to starting point (initial particle sample), slight sensitivity to learning rate and very strongly kernel-dependent.

4 Conclusions

Non-parametric variational methods offer an alternative way to learn about un-normalized distributions in an almost deterministic way through the minimization of a suitable optimization problem. Both SVN (Gauss-Newton) and SVGD serve as algorithms for finding local minima at a cost of $O(n^2)$ per iteration.

The 2-dimensional cases seem to give plausible reasons as to what algorithm is better than the other. But it is not yet very well understood how does these local minima in KL divergence relate to more relevant tasks in estimation such as Moment estimation or rare event probability estimation. We recommend reading [Chen et al., 2018] for more reference about how KL minimization relates to Wasserstein distance and hence moments.

From the computational side, it might be worth exploring different ways of computing the pairwise kernel, which is the bottle in both algorithms, via Fast multipole or other fast structured matrices algorithms.

References

[Chen et al., 2018] Chen, W. Y., Mackey, L., Gorham, J., Briol, F.-X., and Oates, C. J. (2018). Stein points. *arXiv preprint arXiv:1803.10161*.

[Detommaso et al., 2018] Detommaso, G., Cui, T., Marzouk, Y., Scheichl, R., and Spantini, A. (2018). A stein variational newton method. *arXiv preprint arXiv:1806.03085*.

[Liu and Wang, 2016] Liu, Q. and Wang, D. (2016). Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, pages 2378–2386.

[Ruder, 2018] Ruder, S. (2018). An overview of gradient descent optimization algorithms.