ELECTROSTATICS-BASED PARTICLE SAMPLING AND APPROXIMATE INFERENCE

Yongchao Huang
Department of Computing Science
University of Aberdeen, UK
{yongchao.huang}@abdn.ac.uk

ABSTRACT

A new particle-based sampling and approximate inference method, based on electrostatics and Newton mechanics principles, is introduced with theoretical ground, algorithm design and experimental validation. This method simulates an interacting particle system (IPS) where particles, i.e. the freely-moving negative charges and spatially-fixed positive charges with magnitudes proportional to the target distribution, interact with each other via attraction and repulsion induced by the resulting electric fields described by Poisson's equation. The IPS evolves towards a steady-state where the distribution of negative charges conforms to the target distribution. This physics-inspired method offers deterministic, gradient-free sampling and inference, achieving comparable performance as other particle-based and MCMC methods in benchmark tasks of inferring complex densities, Bayesian logistic regression and dynamical system identification. A discrete-time, discrete-space algorithmic design, readily extendable to continuous time and space, is provided for usage in more general inference problems occurring in probabilistic machine learning scenarios such as Bayesian inference, generative modelling, and beyond.

1 INTRODUCTION

Probabilistic approaches Ghahramani (2015); Jordan et al. (1999) are used in many learning and decision-making tasks, e.g. Bayesian inference Winkler (1972); Gelman et al. (2013) and generative modelling Murphy (2022); Bis (2023), and many of them involve estimating the distribution of some quantities of interest. For example, generative adversarial network (GAN) Goodfellow et al. (2014), variational autoencoder (VAE) Kingma & Welling (2013), and more recently diffusion models Song et al. (2020); Ho et al. (2020), either explicitly or implicitly learn some distributions of underlying data or variables. However, in many cases the target density is not straightforwardly approachable, e.g. the posterior in a Bayesian model can be analytically intractable Murray et al. (2006) due to e.g. complex, high-dimensional integration. Bayesian neural networks (BNNs) Ghahramani (2016), for example, issue high-dimensional parameters whose posterior distributions are in general intractable. In such cases, one may resort to approximate inference methods such as Markov chain Monte Carlo (MCMC) MacKay (1998; 2002) or variational inference (VI) methods Blei et al. (2016); Fox & Roberts (2012). These methods perform inference via sampling or optimisation routines.

Sampling from a prescribed distribution, either fully or partially known, is not easy. If we know the full pdf, several statistical methods are available, varying in the resulting sample quality though, for producing (quasi) samples. These include inverse transform sampling Devroye (1986), Box-Muller transform Box & Muller (1958), Latin hypercube sampling (LHS) McKay et al. (1979), rejection sampling von Neumann (1951), importance sampling Kahn & Harris (1949), Quasi-Monte Carlo Methods (QMC) Niederreiter (1992), etc. The most flexible approaches, particularly for sampling from partially known densities, are based on Markov chains Fitzgerald (1996). For example, MCMC methods such as Metropolis–Hastings (MH) sampling Metropolis et al. (1953), Gibbs sampling Geman & Geman (1984), Langevin Monte Carlo (LMC) Roberts & Tweedie (1996), Hamiltonian Monte Carlo (HMC) Duane et al. (1987), slice sampling Neal (2003), etc. VI methods such as mean-field VI Jordan et al. (1999), stochastic VI Hoffman et al. (2013), black-box VI Ranganath et al. (2014), variational autoencoders (VAEs) Kingma & Welling (2013), VI with normalizing flow Rezende & Mohamed (2015), by its nature are designed for inference and widely used in many

applications Jordan et al. (1999); Roberts & Penny (2002); some particle-based VI (ParVI) methods such as Stein variational gradient descent (SVGD) Liu & Wang (2016), Sequential Monte Carlo (SMC) Doucet et al. (2001), particle-based energetic VI (EVI) Wang et al. (2021), etc, also generate samples for inference. MCMC and VI methods together form the paradigm of approximate inference in the Bayesian context; they differ in that, MCMC draws and uses samples for inference while VI in general casts the inference problem into optimisation. Of these sampling algorithms, some utilise the original pdf or up to a constant (i.e. *gradient-free*), e.g. slice sampling and the MH method, while others, e.g. HMC, LMC and most VI methods, utilise the gradient information to guide sampling (i.e. *gradient-based*). Using only gradients enables sampling and inference from an un-normalised, intractable densities which are frequently encountered in Bayesian inference; however, it also increases computational burden.

While MCMC is flexible, can handle multi-modal distributions and provides exact samples in the limit, it is computationally intensive and converge slowly due to undesirable autocorrelation between samples Liu et al. (2019b), they may also require careful tuning and can suffer from convergence issues. It is often impractical for very large datasets or models with complex structures. VI is deterministic, faster and more scalable, but it can struggle with optimization challenges and introduces approximation bias, and may be limited by the expressiveness of the variational family. Both methods, if not all, require estimating the gradients for sampling or optimisation, which can be expensive. Stochastic gradient estimation can be used but extra care should be taken to reduce variance.

ParVI methods enjoy the key strengths of sampling and VI methods: being fast, accurate, deterministic[1], capable of handling multi-modal and sequential inference. ParVI methods evolve a finite set of particles whose terminal configuration upon convergence approximates the target distribution. The particles are deterministically updated using gradients of an explicit or implicit variational objective (e.g. KL-divergence, ELBO, kernelised Stein discrepancy), yielding flexible and accurate approximation Liu et al. (2019b). Compared to MCMC and other VI methods, ParVI offers appealing time-accuracy trade-off Saeedi et al. (2017): the non-parametric nature of ParVI gives greater flexibility than classical VIs, while the interaction between finite particles, along with deterministic optimisation, makes them more efficient than MCMC methods. While most currently available ParVI methods require use of gradients, gradient-free ParVI methods have not been explored much.

Here we introduce a physics-based ParVI method, *EParVI*, which simulates an electrostatics-based interacting particle system (IPS) for sampling and approximate inference. It is deterministic and gradient-free, and has traits of non-parametric flexibility, expressiveness, and approximation accuracy, as well as simplicity. Unlike classic VI methods, it only requires the target density to be queryable up to a constant. As such, the paper is presented as follows: Section.2 illustrates the methodology, experimental verification follows in Section.3; , conclusion and future work in Section.4. Readers are referred to Appendices for more details, e.g. related work and further discussions.

2 Electrostatics-based sampling

The basic idea is simple and rooted in elementary electrostatics: if we place a electric charge, either negative or positive, in a free space without friction [2], it generates an electric field which induces attracting or repulsive forces on other charges. If a set of charges of equal value and same sign, all negative for example, are placed, at steady-state these charges will distribute homogeneously and pairwise distances are maximized by repulsion. This uniform density generation motivates the design of MC-style integration algorithms Hanson (2005), and forms a basic assumption for electrostatic halftoning Schmaltz et al. (2010). If we further place a set of positive charges at fixed locations (e.g. evenly-spaced grids), with pre-assigned magnitudes specified (e.g. proportional to) by a desired distribution (e.g. distribution of grey values in an image) over the space, the initial steady state breaks and a new equilibrium will be established due to repulsion and attraction. Distribution of the negative charges at the new equilibrium, as we shall see later, is exactly the desired distribution. Based on this mechanism, we can assign a set of positive charges at fixed locations with density proportional to a target distribution, and evolve a set of free, negative charges to form the target geometry. The new steady-state distribution of the negative charges is termed the *energetically ideal particle distribution*

[1] Stochastic gradients can be introduced though, see e.g. Yang et al. (2023).

[2] We restrict the discussion within a free space (i.e. a vacuum), which simplifies the physics without losing generality.

Schmaltz et al. (2010). At the new equilibrium, these negative charges serve as samples drawn from the target distribution and can be used to produce, e.g. summary statistics and predictive uncertainties. See Fig.1 for an illustration.

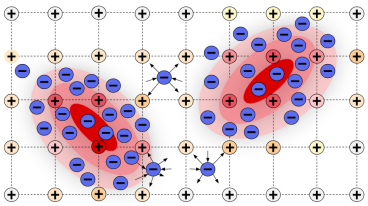


Figure 1: Illustrative diagram of particle distributions. Red darkness indicates the density value of two-Gaussian mixture, arrows denote example forces. Positive charges are fixed at grid points with magnitude proportional to density value, negative charges can move freely.

The central concept used in our electrostatics-based particle sampling and inference algorithm is the interaction between charges (particles) which is governed by the interactive forces. In the following, we first describe the design principles for particle interactions, i.e. the designation of repulsive and attracting forces. Based on these design, we devise an algorithm for simulating this evolving particle system towards the target distribution. Some background knowledge, along with the detailed derivations and results, e.g. 1D, 2D and 3D electric fields and forces, are presented in Appendix.C.

Repulsion To mathematically formalise this idea, we first define the set of *negative* charges [3] (interchangeably used with *particles*) $\{q_i | i = 1, 2, ..., M^{neg}\}$, where M^{neg} it the cardinality of this set, in d-dimensional space, and each is associated with position $\mathbf{x}_i$. Consider a single point charge q_i, let $\mathbf{E}_{q_i}$ be the electric field induced by q_i. In electrostatics, a electric field $\mathbf{E}$ is proportional to the electric displacement field and equals the negative gradient of the electric potential φ; the divergence of $\mathbf{E}$, which implies the infinitesimal electric flux, is linked to the distribution of a source via the *Poisson's equation* for electrostatics (see Eq.10, Eq.14, Eq.15 in Appendix.C). We assume no external source or sink in this charge particle system, which simplifies the Poisson's equation to *Laplace's equation*; solving Laplace's equation gives the potential field φ, and therefore the electric field $\mathbf{E}$ and force $\mathbf{F}$ induced by a point charge (see Appendix.C and Appendix.D for details). The repulsive force between two negative point charges q_i and q_j, locating at a distance of $r_{i,j}$, as per Coulomb's law, is:

$$\mathbf{F}^{rep}_{i \to j}(r_{i,j}) = q_j \mathbf{E}_i(r_{i,j}) = \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2} \varepsilon_0 r_{i,j}^{d-1}} \mathbf{e}_{i \to j} \tag{cc.Eq49}$$

where $\mathbf{E}_i(r_{i,j})$ is the electric field at a distance of $r_{i,j}$ from a inducing point charge q_i. The unit vector $\mathbf{e}_{i \to j} = (\mathbf{x}_j - \mathbf{x}_i)/r_{i,j}$ gives the out-pointing direction. According to *Newton's third law*, if we treat q_j as a source point and q_i a field point, q_i experiences the same amount of force with reverse direction $\mathbf{e}_{j \to i} = -\mathbf{e}_{i \to j}$.

Now we look at the particle set $\{q_i | i = 1, 2, ..., M^{neg}\}$, and consider the combined repulsive force that particle j receives from all other particles, which can be calculated as:

$$\mathbf{F}^{rep}_j(\mathbf{x}_j) = \sum_{i=1, i \neq j}^{M^{neg}} \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2} \varepsilon_0 r_{i,j}^{d-1}} \mathbf{e}_{i \to j} = \sum_{i=1, i \neq j}^{M^{neg}} \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2} \varepsilon_0} \frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|_2^{d/2}} \tag{Eq49b}$$

where $\|\cdot\|_2$ denotes the L_2-norm of a vector. When charges (particles) achieve steady-state with uniform distribution, for each particle $j \in \{1, 2, ..., M^{neg}\}$, the combined repulsive force $\mathbf{F}^{rep}_j(\mathbf{x}_j)$ becomes zero, as all induced electric fields acting on the particle cancel.

One can conveniently re-write the first half of Eq49b in matrix form, by denoting the row vector $\mathbf{q}_j = [q_j, q_j, \cdots, q_j]^T$, column vector $\mathbf{q}_{1:M^{neg}} = [q_1, q_2, \cdots, q_{M^{neg}}]$, diagonal matrix $\mathbf{\Lambda} = diag[1/r_{1,j}^{d-1}, 1/r_{2,j}^{d-1}, ..., 1/r_{i,j}^{d-1}]$, as follows:

$$\mathbf{F}^{rep}_j(\mathbf{x}_j) = \frac{\Gamma(\frac{d}{2})}{2\pi^{d/2} \varepsilon_0} \mathbf{q}_j \mathbf{\Lambda} \mathbf{q}_{1:M^{neg}} \tag{Eq49c}$$

[3] Negative charges are implied by the superscript *neg*, while positive charges by *pos*.

in which $\mathbf{\Lambda}$ can be viewed as some kind of isotropic covariance matrix produced by a kernel (similar to a RBF kernel but not exactly), with each diagonal element measuring a pairwise correlation or inverse-distance between the two point charges q_j and q_i.

Attraction The overall attracting force, resulted from a positive point charge and acting on a negative charge q_j, is Schmaltz et al. (2010):

$$\mathbf{F}_j^{attr}(\mathbf{x}_j) = -\sum_{i'=1,i'\neq j}^{M^{pos}} \frac{qp(\mathbf{x}_{i'})q_j\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0 r_{i',j}^{d-1}}\mathbf{e}_{i'\to j} = -\sum_{i'=1,i'\neq j}^{M^{pos}} \frac{qp(\mathbf{x}_{i'})q_j\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0}\frac{\mathbf{x}_j-\mathbf{x}_{i'}}{\|\mathbf{x}_j-\mathbf{x}_{i'}\|_2^{d/2}} \quad \text{(Eq49d)}$$

where we have set the positive charge $q_j = qp(\mathbf{x}_j)$ with q being a constant scaling factor specifying the relative strength of attracting forces. A negative sign implies attracting force.

If we ignore the contributing forces from those located far away, in regions with constant probability (and therefore uniform positive charge magnitudes), the attracting forces cancel and repulsion still yields the energetically ideal positions which maximize pairwise distances. In regions with uneven probability distribution, attracting force dominates and binds particles onto locations with higher probability. If mesh grid is set to be infinitely small, M^{pos} grows to infinitely large and the distribution of positive charge magnitude becomes continuous, Eq49d becomes an integral.

Evolving the particle system Summation of the repulsive and attracting forces gives the overall force $\mathbf{F}_j = \mathbf{F}_j^{rep} + \mathbf{F}_j^{attr}$, acting on a single free-moving negative charge j; this combined force drives the negative charge to move as per *Newton's second law*. At equilibrium (steady-state), the repulsive force equals the attractive force (in magnitude) and $\mathbf{F}_j$ vanishes. As we intend to evolve the particle system over time, we introduce the time dimension and denote the time-dependent quantities with superscript t. Further, avoiding complex physics, we assume variation of particle displacement $s_{q_j}^t$ is proportional to the corresponding overall acting force $\mathbf{F}_j^t$, i.e. $\delta s_j^t \propto \mathbf{F}_j^t$, which gives the *Euler* position updating rule Schmaltz et al. (2010) for particle j:

$$\mathbf{x}_j^{t+1} = \mathbf{x}_j^t + \tau\mathbf{F}_j^t = \mathbf{x}_j^t + \tau\Big[\sum_{i=1,i\neq j}^{M^{neg}} \frac{q_i q_j\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0}\frac{\mathbf{x}_j-\mathbf{x}_i}{\|\mathbf{x}_j-\mathbf{x}_i\|_2^{d/2}} - \sum_{i'=1,i'\neq j}^{M^{pos}} \frac{qp(\mathbf{x}_{i'})q_j\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0}\frac{\mathbf{x}_j-\mathbf{x}_{i'}}{\|\mathbf{x}_j-\mathbf{x}_{i'}\|_2^{d/2}}\Big] \tag{1}$$

where $\mathbf{F}_j^t$ is the combined force (Eq.54) acting on charge j at time t. τ is a small constant (i.e. step size). Note the index $i \in \{1, 2, ...M^{neg}\}$ indexes the negative charge group, while $i' \in \{1, 2, ...M^{pos}\}$ is for the positive charge group, and if computation allows, the positive charge group can be made continuous over the space.

Assuming a linear relation between force and displacement violates Newton's second law $\mathbf{F} = m\mathbf{a}$ where $\mathbf{F}$ is the overall acting force, m the mass and $\mathbf{a}$ the acceleration; however, this linearisation simplifies and accelerates calculation. Theoretically, to recover Newton's second law, the proportionality constant τ should be non-linear and time-variant. Here we take another approach (see Appendix.D) to (approximately) preserve this physical law using finite difference (we drop the index j for now):

$$\mathbf{a} = \frac{d^2\mathbf{x}}{dt^2} \approx \frac{\mathbf{x}^{t+1} - 2\mathbf{x}^t + \mathbf{x}^{t-1}}{\Delta t^2} \tag{2}$$

with unity mass $m = 1$, we obtain $\mathbf{F} = \mathbf{a}$, leading to the *Verlet* position updating rule:

$$\mathbf{x}_j^{t+1} \approx \mathbf{F}_j^t\Delta t^2 + 2\mathbf{x}_j^t - \mathbf{x}_j^{t-1} = \mathbf{x}_j^t + \mathbf{F}_j^t\Delta t^2 + \Delta\mathbf{x}_j^t \tag{1b}$$

with $\Delta\mathbf{x}_j^t = \mathbf{x}_j^t - \mathbf{x}_j^{t-1}$. We can also design a damped version with a discount factor τ':

$$\mathbf{x}_j^{t+1} = \mathbf{x}_j^t + \tau'\Delta\mathbf{x}_j^t = \mathbf{x}_j^t + \tau'[\mathbf{F}_j^t\Delta t^2 + \Delta\mathbf{x}_j^t] \tag{1c}$$

where $\mathbf{F}_j^t$ remains the same as in Eq.1. Both Eq.1b and Eq.1c require Δt^2 to be sufficiently small to be accurate. The newly introduced hyper-parameter Δt^2 balances the relative importance of the contribution from current step force $\mathbf{F}_j^t$ and the historical displacement $\Delta\mathbf{x}_j^t$ carried over from previous step. This momentum-like updates, hopefully, will give higher fidelity in suggesting next particle position. Note the time discretisation size Δt, along with the step size τ', may impact accuracy and speed of convergence.

Now we have 3 rules, i.e. Eq.1, Eq.1b and Eq.1c, for updating the negative charge positions in any iteration. Simple Euler method may require very small time steps to converge accurately, which leads to slow simulation. The Verlet integration method is commonly used in molecular dynamics. It takes into account the previous position, potentially leading to smoother trajectories; however, it requires more storage as well. The damped Verlet integration provides additional damping but also has more parameter to tune. Eq.1s define a set of *discrete-time*, *discrete-space* formulas for updating particle positions. In order to estimate next particle position, we only need to record the historical positions $\{\mathbf{x}^t, t = 1, 2, ..., k\}$ and calculate the current time overall force $\mathbf{F}_j^t$. We keep updating particles' positions until converge (indicated by either static or small movements). When attaining equilibrium, we have $\mathbf{F}_j^t = 0$ and $\Delta\mathbf{x}_j^t = 0$ for all $j \in \{1, 2, ..., M^{neg}\}$. The charge system then becomes electrically neutral. As the number of negative particles attracted to a region is proportional to the magnitudes of nearby positive charges, particle clustering at the steady state shall align with the distribution of the magnitudes and therefore the target distribution. A vanilla implementation of the *EParVI* algorithm is described in Algo.1.

The *EParVI* algorithm is particularly advantageous for sampling and inferring a partially known density $p(\mathbf{x}) = p'(\mathbf{x})/Z$, e.g. an energy model $p(\mathbf{x}) = e^{-f(\mathbf{x})}/Z$ where the normalising constant Z is unknown due to high-dimensional integral: we can use this algorithm to effectively draw samples from $p(\mathbf{x})$ and make inference about its statistics (e.g. multi-modal means and variances), once the numerator is queryable; we only need to set the positive charge magnitudes $qp(\mathbf{x})$ proportional to the queryable term and possibly normalise them. Therefore, in this context we don't distinguish the case where the target density is known up to a constant - we use the same, overloaded notation $p(\mathbf{x})$ for denoting either the queryable term $p'(\mathbf{x})$ or the full density.

In our experiments, for simplicity, we use equally-charged unit negative charges, and make the maximum magnitude for positive charges to be unity as well, i.e. $q_i = q = 1$. Note in this setting, the absolute value of q_i or q is not relevant; they are absorbed into the hyper-parameters τ and/or Δt^2. However, one does have the option to set the positive charge constant q to be multiples of the negative charge magnitude q_i to address the relative importance of attracting force. Also, if the force is too large, the negative charge can jump to and get trapped in a low-density region (as discussed in Section.K), this can be avoided by normalising the force in each iteration. Besides, extra force terms can be added to $\mathbf{F}_j^t$, e.g. a dragging force controlling the negative charge to move within certain grids Schmaltz et al. (2010), which is similar to regularisation used in many objectives. Also, the move $\mathbf{x}_j^{t+1} = \mathbf{x}_j^t + \tau\Delta\mathbf{x}_j^t$ can be made probabilistically, i.e. we can insert an MH step to assess if this move is worthwhile. These extensions are discussed in Section.K.

Computational complexity In each iteration, computing the pairwise forces takes order $\mathcal{O}(m(M^{neg} + M^{pos})^2)$ *FLOP*s, where the number of positive charges $M^{pos} = M^m$, with M being the number of equi-distance grid points in each dimension, grows exponentially with the dimension m. The overall computational cost over T iterations is therefore $\mathcal{O}(Tm(M^{neg} + M^{pos})^2)$. Major computational efforts can be saved by a wise routine, as proposed in Gwosdek et al. (2014), which reduces the efforts for computing the pairwise forces to $(M^{neg} + M^{pos}) \log(M^{neg} + M^{pos})$. Of course, one can reduces the total number of positive charges M^{pos} by designing an irregular grid for assigning the positive charges. As discussed in Appendix.E, some computational efforts can be saved by performing neighbourhood search which identifies near-field particle sets via e.g. *k-d-tree* search, drops contributions from distant particles and reduces the computation, as well as using neural approximation of the force field.

3 Experiments

3.1 Toy examples via EParVI

Gaussian densities Two examples, a unimodal Gaussian and a bimodal Gaussian mixture (mixing proportions 0.7/0.3), are presented in Fig.2. Fig.2(a) starts from a uniform prior which partially overlaps with the target, converges fast and the inferred shape (third from left) aligns well with ground truth (pink contours). Fig.2(b) identifies the two modes and their mixing proportions very well. Unlike many MCMC algorithms which encourage particles to move from low to high density regions (either following the steepest gradient direction or an MH step), we observe that, the interacting particles following the deterministic EParVI algorithm don't restrict themselves to follow the gradient.

Algorithm 1 EParVI: Electrostatics-based ParVI

Inputs: a queryable target density $p(\mathbf{x})$, particle dimension m, initial proposal distribution $p^0(\mathbf{x})$, total number of iterations T, total number of negative M^{neg} and positive $M^{pos} = M^m$ charges (where M is the number of positive charges along each dimension), Euler constant τ, max positive charge magnitude q.
Outputs: particles $\{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^{M^{neg}}$ whose empirical distribution approximates the target $p(\mathbf{x})$.

1. ***Initialise*** particles.
(1) Mesh the space into $M^{pos} = M^m$ equi-distance grids. Query (and optionally normalise) the density values $p(\mathbf{x}_i)$ at all grid points $\mathbf{x}_i, i \in \{1, 2, ..., M^{pos}\}$.
(2) Assign a positive charge with magnitude $qp(\mathbf{x}_i)$ to each grid point $\mathbf{x}_i$.
(3) Draw M^{neg} m-dimensional negative charges $\mathbf{x}^0 = \{\mathbf{x}_j^0\}_{j=1}^{M^{neg}}$ from the initial proposal distribution $p^0(\mathbf{x})$.

2. ***Update*** negative charge positions.
For $t = 0, 1, 2, ..., T-1$, repeat until converge:
(1) Calculate the overall force acting on each negative charge j (see Eq.Eq49b and Eq.Eq49d): $\mathbf{F}_j^t = \mathbf{F}_j^{rep,t} + \mathbf{F}_j^{attr,t}$
(2) Update negative charge positions (Eq.1 as an example): $\mathbf{x}_j^{t+1} = \mathbf{x}_j^t + \tau \mathbf{F}_j^t$

3. ***Return*** the final configuration of negative charges $\{\mathbf{x}_j^T\}_{j=1}^{M^{neg}}$ and their empirical distribution.

Other densities We also test the EParVI algorithm on three other toy densities, i.e. a moon-shaped density, double-banana, a wave density, which are commonly used in other VI benchmark tests Haario et al. (1999); Chen et al. (2018); Liu et al. (2019a); Wang et al. (2021). The results are shown in Fig.3. With different initialisation, the EParVI algorithm is able to transport particles to the target regions while maintaining diversity. Dispersion of particles is secured by the repulsive forces, which helps prevent model collapse.

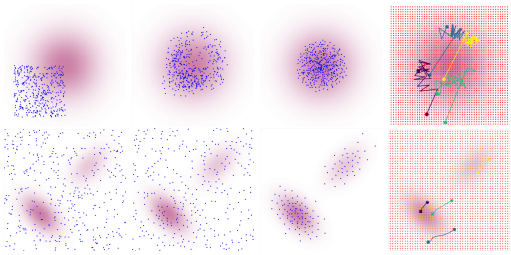


Figure 2: Two toy 2D Gaussian densities. Upper: unimodal, lower: bimodal. Left to right: iteration 0, 5, 60 and selected particle trajectories (dot: start, square: end). Pink contours are ground truth density; blue dots denote negative charges, red dots denote positive charges. Appendix F for details.

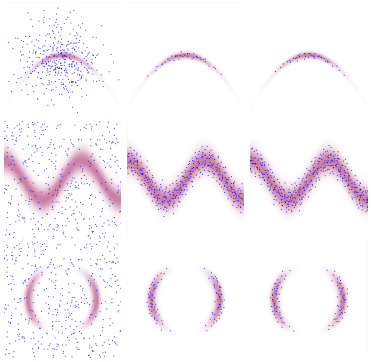


Figure 3: Three other toy 2D densities. Left to right: iteration 0, 40, 60. Appendix F for details.

3.2 Neal's funnel

Neal's funnel distribution, as written in Eq.3, has been challenging to infer due to its highly anisotropic shape and exponentially increasing variance - the funnel's neck is particularly sharp, making it a benchmark for testing many MCMC and VI methods Neal (2003). Neal's funnel specifies $p(x_2) = \mathcal{N}(x_2|0, \sigma^2)$, and $p(x_1|x_2) = \mathcal{N}\big(x_1|0, \exp(x_2/2)\big)$, and their joint pdf

$$p(x_1, x_2) = p(x_1|x_2)p(x_2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x_2^2}{2\sigma^2}\right) \cdot \frac{1}{\sqrt{2\pi \exp(x_2/2)}} \exp\left(-\frac{x_1^2}{2\exp(x_2/2)}\right) \tag{3}$$

We test 6 methods, i.e. MH, HMC, LMC, SVGD EVI and EParVI, on the 2D differentiable geometry with $\sigma = 3$, compare their inference quality, as measured by two metrics, i.e. the squared maximum mean discrepancy MMD^2 Arbel et al. (2019); Wang et al. (2021) and the average negative log-likelihood (*avg NLL*), as well as their runtimes in each iteration. While MH and HMC produce samples sequentially, Langevin, SVGD, EVI and EParVI generate samples iteratively and are started with the same initialisation.

Results presented in Fig.4 show that, with a medium number of particles (i.e. 400), MH and LMC are the fastest in terms of speed, SVGD and EParVI are among the slowest due to large number of iterations between particles. EVI is fastest among these particle methods. HMC, being fast and accurate, is considered as the golden method for sampling complex geometriesHoffman & Gelman (2014); we therefore use the HMC samples as a reference for evaluating MMD^2. Key observations are: while all samples show some similarity to the HMC samples, EParVI samples notably yield the smallest average NLL, which implies they are better quasi samples following the target distribution, as also evidenced by the upper row figures.

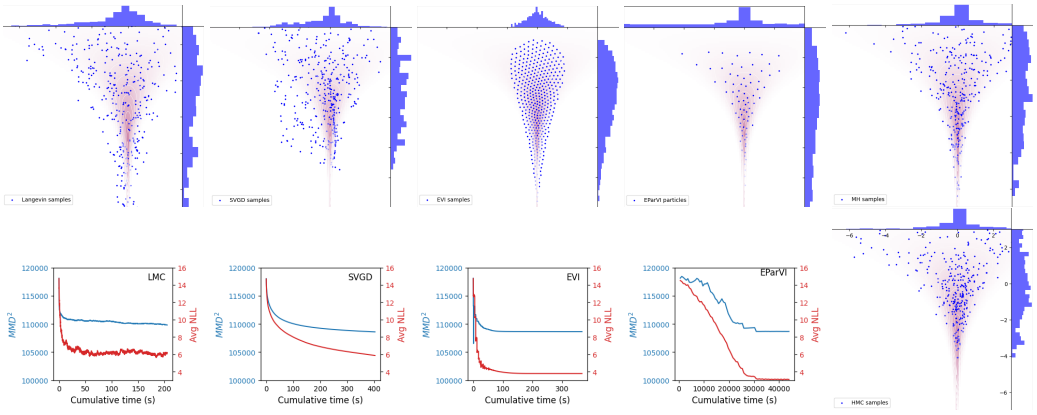


Figure 4: Inference of the Neal's funnel using six methods. Upper left to right: LMC, SVGD, EVI, EParVI, and MH samples. Lower left to right: corresponding MMD^2, NLL values over runtime and HMC samples. MH and HMC take seconds to run. See Appendix.G for details.

3.3 Bayesian Logistic Regression

We also apply the proposed EParVI, along with other 4 counterparts, i.e. logistic regression via maximum likelihood (MLE-LR), HMC, SVGD and EVI, to infer the coefficients of a low-dimensional Bayesian logistic regression (BLR) model, using the Iris dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ which has $N = 150$ observations with binary labels y_i, and each instance $\mathbf{x}_i$ consists of 4 features. The logistic regression model is $p(y_i = 1|\mathbf{x}_i, \boldsymbol{\omega}) = \frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}$. We use standard Gaussian prior for regression coefficients, i.e. $\boldsymbol{\omega} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The posterior distribution of coefficients $\boldsymbol{\omega}$ is then (details please refer to Appendix.H):

$$p(\boldsymbol{\omega}|\boldsymbol{y}, \mathbf{X}) \propto \prod_{i=1}^{N} \left[\frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}\right]^{y_i} \cdot \left[1 - \frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}\right]^{1-y_i} \exp\left(-\frac{1}{2\alpha}\boldsymbol{\omega}^T\boldsymbol{\omega}\right) \quad (4)$$

which is used as the target for our sampling methods.

The initial and final samples are visualised for the two dimensions (ω_1,ω_2) in Fig.5 in Appendix.H. All 4 methods yield reasonably well inference quality - all terminal particles cluster around the ground truth point with some dispersion representing uncertainties. The EParVI particles, in particular, have reduced uncertainty as compared to other methods. A comparison of the statistics are made in Table.1, in which we observe HMC yields results closest to the MLE-LR estimates. Particle methods give less accurate estimates yet fast speeds such as SVGD and EVI. EParVI achieves smallest NLL but less impressive mean estimates; it also differs from the HMC samples due to less iterations and the coarse granularity used in meshing.

Method	$\hat{\omega}_1$	$\hat{\omega}_2$	$\hat{\omega}_3$	$\hat{\omega}_4$	NLL/MMD^2	Runtime
MLE-LR	-0.64	1.89	-1.75	-1.67	NA	$<$ 1 second
HMC	-0.69	1.98	-1.84	-1.86	NA	$<$ 30 seconds
SVGD	-0.76	1.87	-1.74	-1.67	12.83 / 2.41	$<$ 100 seconds
EVI	-0.78	1.81	-1.63	-1.56	11.39 / 10.14	$<$ 100 seconds
EParVI	-0.64	1.62	-1.50	-1.42	11.32 / 23.25	$>$ 10^3 seconds

Table 1: Comparison of inference methods for the Bayesian Logistic Regression (BLR) problem. Sample mean estimates of coefficients are presented for all but LR. With these inferred coefficients, the test accuracy for all cases is 100%. Runtimes are tested on *Colab TPU v2*.

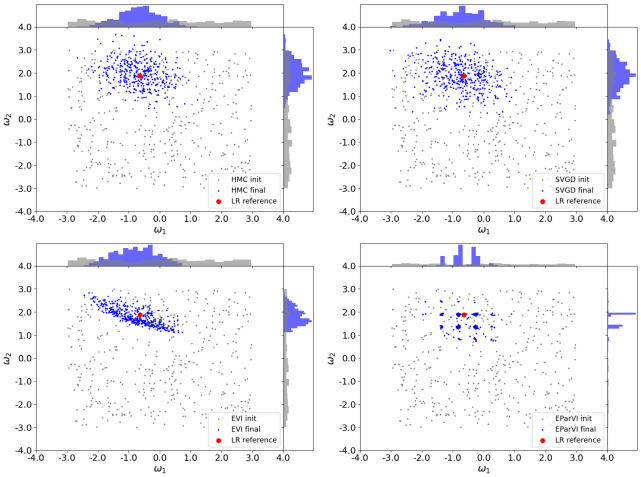


Figure 5: Inference of BLR model. Upper left to right: HMC and SVGD samples. Lower left to right: EVI and EParVI samples. MH and HMC take seconds to run. See Appendix.H for details.

3.4 Inference of a dynamical system

We also compared the performance of MAP, HMC, LMC, SVGD and EParVI methods on inferring the posterior distributions of the parameters in the Lotka-Volterra dynamical system, results are presented in Appendix.I.

4 Conclusion and future work

Conclusion We propose a variational sampling and inference method, EParVI, which adheres to physical laws and exhibits superior performance in inferring low-dimensional, complex geometries; it also achieves comparable performance as other MCMC and VI methods in a real-world Bayesian logistic regression problem. EParVI simulates an interacting particle system in which positive particles are fixed at grid points and negative particles are free to move. With positive charge magnitudes proportional to the target distribution, negative particles evolve towards a steady-state approximating the target distribution; particles repositioning follows a discretised Newton's second law, and the force acts in accordance with the electric fields described by the Poisson's equation. It features gradient-free, deterministic dynamics sampling, with traits of simplicity, non-parametric flexibility, expressiveness and high approximation accuracy; it also suffers from curse of dimensionality.

Future work There are theoretical aspects which can be explored. For example, the convergence of the IPS towards equilibrium with theoretical guarantees. This can be routed to using the variational potential energy objective. Also, improving the computational efficiency is significant. Further, this technique can be potentially used for optimisation. However, sampling differs from optimisation in that, we can query the target function (i.e. the density, possibly partially known) as many times as we want; in optimisation, however, one typically avoids to perform functional evaluations too many times as it's expensive.

REFERENCES

Deep Learning - Foundations and Concepts. Springer International Publishing, 2023. doi: 10.1007/978-3-031-45468-4. URL https://link.springer.com/book/10.1007/978-3-031-45468-4.

Mikołaj Arbel, Ahmed Korba, Arash Salim, and Arthur Gretton. Maximum mean discrepancy gradient flow. In *Advances in Neural Information Processing Systems*, pp. 6484–6494, 2019.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *https://arxiv.org/pdf/1601.00670.pdf*, 2016.

S. L. Bore and F. Paesani. Realistic phase diagram of water from "first principles" data-driven quantum simulations. *Nature Communications*, 14(1):3349, 2023.

George E. P. Box and Mervin E. Muller. A note on the generation of random normal deviates. *The Annals of Mathematical Statistics*, 29(2):610–611, 1958.

Bob Carpenter. Predator-prey population dynamics: the lotka-volterra model in stan, 2022. URL https://mc-stan.org/users/documentation/case-studies/lotka-volterra-predator-prey.html.

Changyou Chen, Ruiyi Zhang, Wenlin Wang, Bai Li, and Liqun Chen. A unified particle-optimization framework for scalable bayesian sampling, 2018.

Daniel Cristofaro-Gardiner. The laplace and the poisson equations in the whole space, 2017. Lecture Notes, Carnegie Mellon University.

Luc Devroye. *Non-Uniform Random Variate Generation*. Springer-Verlag, New York, 1986.

Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, New York, 2001.

Simon Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216–222, 1987. doi: 10.1016/0370-2693(87)91197-X.

Timothy T. Duignan. The potential of neural network potentials. *ACS Physical Chemistry Au*, 0(0):null, 0. doi: 10.1021/acsphyschemau.4c00004. URL https://doi.org/10.1021/acsphyschemau.4c00004.

Joseph J.K. O Ruanaidh; William J. Fitzgerald. *Numerical Bayesian Methods Applied to Signal Processing*, pp. 244. Springer, 1st edition, 1996.

C. W. Fox and S. J. Roberts. A tutorial on variational bayesian inference. *Artificial Intelligence Review*, 38:85–95, 2012. doi: 10.1007/s10462-011-9236-8. URL https://doi.org/10.1007/s10462-011-9236-8.

Borko Furht (ed.). *Dithering*, pp. 198–199. Springer US, Boston, MA, 2008. ISBN 978-0-387-78414-4. doi: 10.1007/978-0-387-78414-4_18. URL https://doi.org/10.1007/978-0-387-78414-4_18.

Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, New York, 3 edition, 2013. doi: 10.1201/b16018. URL https://doi.org/10.1201/b16018.

Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6 (6):721–741, 1984. doi: 10.1109/TPAMI.1984.4767596.

Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553): 452–459, 2015. doi: 10.1038/nature14541.

Zoubin Ghahramani. A history of bayesian neural networks. http://bayesiandeeplearning.org/2016/slides/nips16bayesdeep.pdf, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Pascal Gwosdek, Christian Schmaltz, Joachim Weickert, and Tanja Teuber. Fast electrostatic halftoning. *Journal of Real-Time Image Processing*, 9(2):379–392, 2014. doi: 10.1007/s11554-011-0236-3.

Heikki Haario, Eero Saksman, and Johanna Tamminen. Adaptive proposal distribution for random walk metropolis algorithm. *Computational Statistics*, 14(3):375–396, 1999.

Kenneth M. Hanson. Halftoning and quasi-monte carlo. In Kenneth M. Hanson and Francois M. Hemez (eds.), *Sensitivity Analysis of Model Output*, pp. 430–442. Los Alamos Research Library, 2005. URL `http://library.lanl.gov/ccw/samo2004/`.

Russell Herman. *Introduction to Partial Differential Equations*. LibreTexts, 2020. University of North Carolina Wilmington.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

Matthew D. Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15:1593–1623, 2014. Submitted 11/11; Revised 10/13; Published 4/14.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Matthew D. Homan and Andrew Gelman. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 1 2014. ISSN 1532-4435.

Robert Hunt. Mathematical methods ii (natural sciences tripos, part ib) - poisson's equation. `https://www.damtp.cam.ac.uk/user/reh10/lectures/nst-mmii-chapter2.pdf`, 2002. Lecture notes, Department of Applied Mathematics and Theoretical Physics, University of Cambridge.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

Herman Kahn and Theodore E. Harris. Estimation of particle transmission by random sampling. In *Monte Carlo Method*, volume 12 of *Applied Mathematics Series*, pp. 27–30. National Bureau of Standards, 1949.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

Chang Liu, Jiaxuan Zhuo, Peng Cheng, Ruqi Zhang, and Jun Zhu. Understanding and accelerating particle-based variational inference. In *International Conference on Machine Learning*, pp. 4082–4092, 2019a.

Chang Liu, Jingwei Zhuo, Pengyu Cheng, Ruiyi Zhang, Jun Zhu, and Lawrence Carin. Understanding and accelerating particle-based variational inference. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019b. URL `http://proceedings.mlr.press/v97/liu19d/liu19d.pdf`.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in Neural Information Processing Systems 29 (NIPS)*, pp. 2378–2386, 2016. URL `https://arxiv.org/abs/1608.04471`.

D. J. C. MacKay. Introduction to monte carlo methods. In M. I. Jordan (ed.), *Learning in Graphical Models*, pp. 175–204. Kluwer Academic Press, 1998.

David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, USA, 2002. ISBN 0521642981.

M. D. McKay, W. J. Conover, and R. J. Beckman. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2): 239–245, 1979.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114.

Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL probml.ai.

Iain Murray, Zoubin Ghahramani, and David J. C. MacKay. Mcmc for doubly-intractable distributions. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, UAI'06, pp. 359–366, Arlington, Virginia, USA, 2006. AUAI Press. ISBN 0974903922.

Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–767, 2003. doi: 10.1214/aos/1056562461.

Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, volume 63 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, 1992.

Ryutaroh Ohbuchi and Masaki Aono. Quasi-monte carlo rendering with adaptive sampling. http://www.kki.yamanashi.ac.jp/~ohbuchi/online_pubs/eg96_html/eg96.htm, 1996.

Abril-Pla Oriol, Andreani Virgile, Carroll Colin, Dong Larry, Fonnesbeck Christopher J., Kochurov Maxim, Kumar Ravin, Lao Jupeng, Luhmann Christian C., Martin Osvaldo A., Osthege Michael, Vieira Ricardo, Wiecki Thomas, and Zinkov Robert. Pymc: A modern and comprehensive probabilistic programming framework in python. *PeerJ Computer Science*, 9:e1516, 2023. doi: 10.7717/peerj-cs.1516.

Rajesh Ranganath, Sean Gerrish, and David M. Blei. Black box variational inference. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 814–822, 2014. URL http://proceedings.mlr.press/v33/ranganath14.pdf.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp. 1530–1538, 2015. URL https://arxiv.org/abs/1505.05770.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016a. URL https://arxiv.org/abs/1505.05770.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016b.

Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996. doi: 10.2307/3318418.

Stephen J Roberts and Will D Penny. Variational bayes for generalized autoregressive models. *IEEE Transactions on Signal Processing*, 50(9):2245–2257, 2002. doi: 10.1109/TSP.2002.802091.

Tetsuya Kaji; Veronika Ročková. Metropolis-hastings via classification. *https://arxiv.org/pdf/2103.04177.pdf*, 2021.

Ardavan Saeedi, Tejas D. Kulkarni, Vikash K. Mansinghka, and Samuel J. Gershman. Variational particle approximations. *Journal of Machine Learning Research*, 18:1–29, 2017.

Christian Schmaltz, Pascal Gwosdek, Andrés Bruhn, and Joachim Weickert. Electrostatic halftoning. *Computer Graphics Forum*, 29(8):2313–2327, 2010. doi: 10.1111/j.1467-8659.2010.01716.x.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *CoRR*, abs/2011.13456, 2020. URL https://arxiv.org/abs/2011.13456.

T. Teuber, G. Steidl, P. Gwosdek, C. Schmaltz, and J. Weickert. Dithering by differences of convex functions. *SIAM Journal on Imaging Sciences*, 4(1):79–108, 2011. doi: 10.1137/100790197. URL https://doi.org/10.1137/100790197.

John von Neumann. Various techniques used in connection with random digits. *Applied Math Series, Notes by G. E. Forsythe, in National Bureau of Standards*, 12:36–38, 1951.

Dilin Wang, Ziyang Tang, Chandrajit Bajaj, and Qiang Liu. Stein variational gradient descent with matrix-valued kernels, 2019.

Yiwei Wang, Jiuhai Chen, Chun Liu, and Lulu Kang. Particle-based energetic variational inference. *Statistics and Computing*, 31(34), 2021. doi: 10.1007/s11222-021-10009-7. URL https://doi.org/10.1007/s11222-021-10009-7.

Robert L. Winkler. *An Introduction to Bayesian Inference and Decision*. Holt McDougal, 1972.

Yifan Yang, Chang Liu, and Zhengze Zhang. Particle-based online bayesian sampling. *ArXiv*, abs/2302.14796, 2023. URL https://api.semanticscholar.org/CorpusID:257232350.

A ELECTROSTATIC HALFTONING

In physics, electrostatics studies the electromagnetic phenomena and properties of stationary or slow-moving electric charges, i.e. when a static equilibrium [4] has been established. Digital image halftoning, on the other hand, concerns about the distribution of colors Furht (2008). Image halftoning algorithms have been developed for sampling problems which often occur in rendering, relighting, printing, object placement, and image visualisation Schmaltz et al. (2010). Electrostatics-based methods are first introduced to image processing Schmaltz et al. (2010); Gwosdek et al. (2014) in which *electrostatic halftoning*, a similar approach to ours, was devised to sample image pixels based on its grey values. Digital images are represented by discrete pixels on a computer (though may not be visually visible) and rendered to finite resolution in tasks such as display and printing Hanson (2005). This requires (re)creating the illusion of e.g. a grey-scale image by distributing the black dots on a white background (i.e. *digital halftoning* [5] Hanson (2005)), or resampling the image to adapt to machine resolution, whilst yielding either energetically or visually optimal results Schmaltz et al. (2010). To accurately and flexibly represent the original image, Schmaltz et al. Schmaltz et al. (2010) proposed the *electrostatic halftoning* method which follows the physical principles of electrostatics to assign and move negative charges, by interacting with positive charges representing image intensities, to form the distribution of grey values. This method is based on the intuitive assumption that, in a region of constant intensity, black points (represented by negative charges) in the sampled images should be equally distributed, while in other regions, through a process of particle interaction and evolution, their distribution will be proportional to the image grey values.

We therefore note the similarity between (standard and quasi) Monte Carlo sampling and digital halftoning: while both methods aim to resemble a target geometry [6], they also share the same problem of sample clumpiness Hanson (2005) and empty spots Ohbuchi & Aono (1996), i.e. undesirable clumping of samples (dots in an image) which degrades sampling efficiency or image quality (uneven dot placement that accompanies random dot distributions introduces undesirable texture).

B RELATED WORK

Using electrostatics principles to model interactions in a particle system is less exposed but not new. In classic Newton mechanics, the interactions between M bodies is described the Newton's third law

[4] When there are no moving charges, the particle system is said to have achieved a static equilibrium, i.e. the vector sum of all forces acting upon the particle is zero.

[5] We only discuss in the context of digital halftoning which renders gray-scale images (i.e. putting black dots on a white surface), and mainly concern about rendering speed and quality.

[6] In an inference task, MC methods try to infer the target density; digital halftoning creates a pattern of black dots to replicate the distribution of image intensity.

$\mathbf{F}_i = -\mathbf{F}_j, i \neq j, i, j \in \{1, 2, ..., M\}$, and each individual force follows the Newton's second law $\mathbf{F} = m\mathbf{a}$ where m is mass and $\mathbf{a}$ acceleration. Gravitational force, for example, can be specified as per the inverse-square law: $\mathbf{F} = G\frac{m_1 m_2}{r^2}$ where r is the distance between the two interacting objects, and G the gravitational constant. These laws can be derived in a variational form via the energy perspective, i.e. the Maxwell equations (a set of coupled PDEs describing electromagnetic laws), or the Gauss law (which in electrostatics relates the distribution of electric charge to the resulting electric field), the Laplace equation (see Appendix.C) and the Poisson's equation (generalisation of Laplace's equation) in more specific domains such as mathematics and physics. Regarding a set of charges as particles and modelling their interactions using already-known electrostatic principles requires an alignment between the the behaviors of the particle system we aim to model and that of the charges. In particle-based variational inference (ParVI), the SVGD method Liu & Wang (2016) for example, particles interact and dynamically move towards forming the target geometry; in the charge system, positive and negative charges attract and repel each other and achieve certain distribution at equilibrium. Although we don't intend to binarily classify particles in ParVI (which is essential in electrostatics), the similarity between both inspires the design of new algorithms at their intersection, by taking advantages of both worlds.

A straightforward application of the interacting electrostatic particle system emerged in digital image processing Ohbuchi & Aono (1996); Hanson (2005); Schmaltz et al. (2010). Inspired by principles of electrostatics, Schmaltz et al. Schmaltz et al. (2010) proposed the *electrostatic halftoning* method for image dithering, stippling, screening and sampling (more details see Appendix.A). This physics-based image processing method is simple, intuitive and flexible: inkblots are modelled as small, massless particles with negative unit charge, while the underlying image is modelled as a positive charge density, the attracting forces attract negative charges move proportionally towards dark image regions. Eventually, the particle system reaches an equilibrium state that forms an accurate halftone representation of the original image. However, their vanilla implementation has a runtime complexity of $\mathcal{O}(M^2)$ in the number of particles M, making it impractical for real-world applications (e.g. high-resolution image halftoning). Gwosdek et al. Gwosdek et al. (2014) proposed a GPU-parallel algorithm to improve the computational efficiency of the electrostatic halftoning method. They simplified the pairwise force calculation by evaluating the near-field and far-field interactions using different strategies, e.g. a novel nearest-neighbour identification scheme was designed, as well as a GPU-based, non-equispaced fast Fourier transform (NFFT) algorithm devised for evaluating interactions between distant particles, which reduces the runtime complexity to $\mathcal{O}(M \log M)$.

Albeit similarities exist between MC sampling and halftoning patterns, the intersection between MC and halftoning techniques has not received much attention in the past Ohbuchi & Aono (1996); Hanson (2005). Ohbuchi et al. Ohbuchi & Aono (1996) applied the QMC method, which employs deterministic low-discrepancy sequences (LDSs, a screen pixel sampling method), to solve the global illumination problem in 3D computer graphics. They used QMC sequences to improve rendering of surfaces and shadows in a 3D scene, and showed that, QMC integral with LDSs converges significantly faster than MC integral and yielded more accurate results. This QMC-LDS method also allows for incremental sampling. Inspired by halftoning, Hanson Hanson (2005) designed a Quasi-Monte Carlo (QMC) method for generating random points sets for evaluating 2D integrals, which reduces the number of function evaluations and yields more accurate estimates than standard QMC methods. Although computation is cheap, however, this method is inapplicable to dimensions larger than four or five as it relies on the calculation of the dimension-restricted mean-squared error. This haltoning-inspired QMC method can also be applicable to sensitivity analysis of computer models. They also proposed a potential-field approach, which considers the repulsion between particles as well as between particles and the boundaries, to obtain 2D random, quasi-uniform integration points.

Our contributions We re-invent the idea of *electrostatic halftoning* Schmaltz et al. (2010) for modelling the interacting particle system (IPS), and apply it to sample and infer probability densities. Although the physics is simple, and the idea of modelling IPS using electrostatics already exists, this work is novel in following three aspects: (1) it is for the first time principles of electrostatics and Newton mechanics are introduced to variational inference; (2) simplified IPS modelling; (3) we extend its theoretical applicability into arbitrary dimensions. The repulsion-attraction framework is different from previous work e.g. Schmaltz et al. (2010) in that, we generalise the detailed physics to any dimension, and it is more flexible to set the parameters, e.g. charge magnitudes with an annealing scheme, by relaxing the equilibrium condition for accelerating convergence. We also introduce two new particle updating rules, rather than assuming linear relation between force and displacement, for

yielding more accurate approximation. Other tricks such as particle filtering, noise perturbation and probabilistic move are also introduced.

C A HYPOTHETICAL TREATMENT OF COULOMB'S LAW IN d-DIMENSION

Here we aim to derive the forces between two charges in low and high dimensions. We first present the well-known force formula in 3D (Section.C.1), then move to general *d* dimensions (Section.C.2), solve the 3D Poisson's equation (Section.C.3), and solve the Poisson's equation in *d* dimensions (Section.C.4).

C.1 ELECTRIC FORCES IN 3D

In three-dimensions, the electric force between two charges q_i and q_j is calculated as per *Coulomb's law*:

$$\mathbf{F} = k\frac{q_i q_j}{r^2}\mathbf{e} \tag{5}$$

where r is the distance between the two charges, $k = 10^{-7}c^2 = \frac{1}{4\pi\varepsilon_0}$ is the *Coulomb constant* (also called the electric force constant or electrostatic constant), with c being the light speed in a vacuum, ε_0 the *vacuum permittivity* introduced later. The unit vector **e** implies the direction of the forces - they point in inverse directions as per *Newton's third law*. The force is repulsive for like and the attracting for unlike electric charges. Note that, Eq.5 only holds for three dimensions in which both Coulomb's law and Newton's law of gravity takes the form of distance inverse-squared.

The electric field, e.g. induced by the charge q_i, is defined as:

$$\mathbf{E}_i = k\frac{q_i}{r^2}\mathbf{e} \tag{6}$$

which conveniently simplifies Eq.5 to $\mathbf{F} = q_1\mathbf{E}_2$.

Superposition principle applies to the electromagnetic field. The overall electric field and force acting on a particle, induced by others, can be calculated by summing over all the contributions due to individual source particles. Consider a collection of M particles each with charge q_i and distribute in a 3D space. The electric field at arbitrary point (other than the point itself) can be expressed as:

$$\mathbf{E}(r) = \frac{1}{4\pi\varepsilon_0}\sum_{i=1}^{M}\frac{q_i}{r_i^2}\mathbf{e}_i \tag{7}$$

where r_i is the distance between a source point and the point of consideration. $\mathbf{e}_i$ is a unit vector indicating the direction of the electric field.

The electric field generated by a distribution of charges, as given by the charge density $\rho(\mathbf{r})$, can be obtained by replacing the sum by integral:

$$\mathbf{E}(r) = \frac{1}{4\pi\varepsilon_0}\iiint\frac{\rho(\mathbf{r}_i)}{r_i^2}\mathbf{e}_i d^3\mathbf{r} \tag{8}$$

which accounts for all contributions from the small charges $\rho(\mathbf{r}_i)d^3\mathbf{r}$ contained in the infinitesimally small volume element $d^3\mathbf{r} \approx dxdydz$.

All these equations are only valid for three dimensions; ideally, we would like to generalize Coulomb's law for *d*-dimension. In the following, we use Poisson's equation (and Laplace's equation) for deriving the electric force in *d* dimensions.

C.2 THE POISSON AND LAPLACE EQUATIONS FOR ELECTROSTATICS

We start with the Gauss's law for electricity in differential form:

$$\nabla \cdot \mathbf{D} = \rho \tag{9}$$

where $\nabla\cdot$ is the divergence operator [7]. $\mathbf{D}$ is the electric displacement field, ρ is the free-charge density (charges from outside). Gauss's law is one of Maxwell's equations; it relates the distribution of electric charge to the resulting electric field.

Assuming linear, isotropic and homogeneous medium, the following constitutive equation holds:

$$\mathbf{D} = \varepsilon \mathbf{E} \tag{10}$$

where $\mathbf{E}$ is the electric field. ε is the absolute permittivity [8] of the medium. Often, the relative permittivity, which is the ratio of the absolute permittivity ε and the vacuum permittivity ε_0, i.e. $\varepsilon_r = \varepsilon/\varepsilon_0$, is used. The vacuum permittivity [9] is the absolute dielectric permittivity of classical vacuum; it is referred to as the *permittivity of free space* or the *electric constant* whose value $\varepsilon_0 \approx 8.8541878128 \times 10^{-12} F/m$ (farads per meter).

The electrostatic force $\mathbf{F}$ on a charge q', induced by the electric field $\mathbf{E}$, is the product:

$$\mathbf{F} = q'\mathbf{E} \tag{11}$$

Substituting Eq.10 into Gauss's law in Eq.9, and assume constant permittivity ε of the medium, we have:

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon} \tag{12}$$

In the presence of constant magnetic field or no magnetic field (i.e. if the field is irrotational), we have:

$$\nabla \times \mathbf{E} = 0 \tag{13}$$

where $\nabla\times$ is the curl operator. As the curl of any gradient is zero, we can write the electric field E as the *gradient* of a scalar function φ (the electric potential):

$$\mathbf{E} = -\nabla\varphi \tag{14}$$

where ∇ here denotes gradient, and φ is a real or complex-valued function on a manifold. Eq.14 states that, the gradient of the electric potential gives the electric field. The negative sign reflects that the φ is the electric potential energy per unit charge.

Substituting Eq.14 into Eq.12 gives:

$$\nabla \cdot \mathbf{E} = -\nabla^2\varphi = \frac{\rho}{\varepsilon} \tag{15}$$

where $\nabla \cdot \nabla = \nabla^2 = \Delta$ is the Laplace operator [10]. Eq.15 relates the divergence of the electric field $\nabla \cdot \mathbf{E}$ to the charge density ρ through the negative Laplacian of the potential φ, which further leads to the *Poisson's equation* for electrostatics:

$$\Delta\varphi = \nabla^2\varphi = -\frac{\rho}{\varepsilon} \tag{16}$$

Solving the Poisson's equation Eq.16 gives the electric potential φ, given the mass density distribution of charges $\rho(\mathbf{x})$; however, in order to solve the Poisson's equation, we need to know the charge density distribution ρ. Different configurations of ρ gives different solutions. For example, if the charge

[7] In three-dimensional Cartesian coordinates, for example, the divergence of a continuously differentiable vector field $\mathbf{D} = D_x\mathbf{i} + D_y\mathbf{j} + D_z\mathbf{k}$ is defined as the scalar-valued function: $div\mathbf{D} = \nabla \cdot \mathbf{D} = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}) \cdot (D_x, D_y, D_z) = \frac{\partial D_x}{\partial x} + \frac{\partial D_y}{\partial y} + \frac{\partial D_z}{\partial z}$. The divergence operator in cylindrical and spherical coordinates can be similarly derived.

[8] In electromagnetism, the absolute permittivity ε is a measure of the electric polarizability of a dielectric material. A material with high permittivity polarizes more in response to an applied electric field than a material with low permittivity, thereby storing more energy in the material.

[9] It measures how dense of an electric field is "permitted" to form in response to electric charges, and relates the units for electric charge to mechanical quantities such as force given by Coulomb's law.

[10] Laplace operator is a second-order differential operator in the *n*-dimensional Euclidean space, defined as the divergence ($\nabla\cdot$) of the gradient ($\nabla\varphi$), where φ is a twice-differentiable function. Under Cartesian coordinates, the Laplacian of φ is thus the sum of all unmixed second partial derivatives: $\Delta\varphi(\mathbf{x}) = \sum_{i=1}^{d} \frac{\partial^2\varphi(\mathbf{x})}{\partial x_i^2}$. Similar expressions can be derived in other coordinates (e.g. cylindrical and spherical coordinates).

density $\rho = 0$ (i.e. in a region where there are no charges or currents), which gives the *Laplace's equation* (the homogeneous Poisson's equation):

$$\nabla^2 \varphi = 0 \tag{17}$$

a function satisfying the Laplace equation is called *harmonic* Cristofaro-Gardiner (2017).

A side note on Poisson's and Laplace equations. Poisson's equation, in its general form as in Eq.19, can be used to describe the diffusion process of a chemical solute or heat transfer, i.e. $\partial\varphi(\mathbf{x}, t)/\partial t = \kappa\nabla^2\varphi = \kappa f(\mathbf{x,t})$, where κ is termed the *diffusivity*. The RHS $f(\mathbf{x}, t)$ represents external sources (for example, in Eq.16 we have $f(\mathbf{x}, t) = -\rho/\varepsilon$) which inject inputs into the system (e.g. solute generated by a chemical reaction, or heat provided by external sources). The Laplace equation, as a special case of the Poisson's equation, can be derived as a steady-state condition for the diffusion process (e.g. concentration of chemical solute, heat transfer). The Poisson's equation doesn't have a unique solution, unless extra boundary condition is imposed. For example, if the electrostatic potential $\varphi(\mathbf{x})$ is specified on the boundary of a region, then its solution is uniquely determined. Some application scenarios of the Poisson's equation, solutions, as well as uniqueness proofs, can be found in e.g. Hunt (2002).

We formerly define $\varphi : \mathbb{R}^d \to \mathbb{R}$, and its Laplacian, $\Delta\varphi : \mathbb{R}^d \to \mathbb{R}$, i.e.

$$\Delta\varphi(\mathbf{x}) := \sum_{i=1}^{d} \partial^2_{x_i x_i} \varphi(\mathbf{x}) \tag{18}$$

where x_i is the ith coordinate of $\mathbf{x} \in \mathbb{R}^d$. In crafting Eq.18, we have assumed that the Laplacian extends the same form into d-dimension.

C.3 The 3D Poisson's equation

Without losing generality [11], we start with solving a 3D *Poisson's equation* generalised from Eq.16:

$$\Delta\varphi(\mathbf{x}) = \nabla^2\varphi(\mathbf{x}) = f(\mathbf{x}) \tag{19}$$

where $\varphi : \mathbb{R}^3 \to \mathbb{R}$ and $f : \mathbb{R}^3 \to \mathbb{R}$ (the support can be restricted to subset of $\mathbb{R}^3$ though). The RHS $f(x)$ is called the *source* and it is often zero either everywhere or everywhere but some specific region or points. For example, $f(\mathbf{x}) = c_0\delta^3(\mathbf{x})$, where c_0 is a constant, represents a point source.

We shall apply Fourier transform to Eq.19; before that, we re-cap the 3D Fourier transform:

$$\hat{\varphi}(\mathbf{k}) = \int \varphi(\mathbf{x}) e^{-i\mathbf{kx}} d^3\mathbf{x} \tag{20}$$

where $d^3\mathbf{x} = dx_1 dx_2 dx_3$, $\mathbf{k} = k_{x_1}\mathbf{i} + k_{x_2}\mathbf{j} + k_{x_3}\mathbf{k}$ is a three dimensional wavenumber. The inverse Fourier transform is

$$\varphi(\mathbf{x}) = \frac{1}{(2\pi)^3} \int \hat{\varphi}(\mathbf{k}) e^{i\mathbf{kx}} d^3\mathbf{k} \tag{21}$$

where $d^3\mathbf{k} = dk_{x_1} dk_{x_2} dk_{x_3}$.

Assuming that $\varphi(\mathbf{x})$ and its gradients vanish for large $\mathbf{x}$, we apply Fourier transform to the generalised 3D Poisson's equation (Eq.19):

$$\mathcal{F}[\nabla^2\varphi(\mathbf{x})] = -(k^2_{x_1} + k^2_{x_2} + k^2_{x_3})\hat{\varphi}(\mathbf{k}) = \mathcal{F}[f(\mathbf{x})] = \hat{f}(\mathbf{k}) \tag{22}$$

by defining $k^2 = k^2_{x_1} + k^2_{x_2} + k^2_{x_3}$, we have:

$$k^2\hat{\varphi}(\mathbf{k}) = \hat{f}(\mathbf{k}) \tag{23}$$

which gives

$$\hat{\varphi}(\mathbf{k}) = \frac{1}{k^2}\hat{f}(\mathbf{k}) \tag{24}$$

[11] This subsection mainly follows Herman (2020).

The solution to the generalised 3D Poisson's equation (Eq.19) can be obtained by performing inverse Fourier transform:

$$\varphi(\mathbf{x}) = \frac{1}{(2\pi)^3} \int \hat{f}(\mathbf{k}) \frac{e^{i\mathbf{kx}}}{k^2} d^3\mathbf{k} \tag{25}$$

As a side note, one can also apply inverse Fourier transform to Eq.22, as a pathway for evaluating the Laplacian $\nabla^2\varphi(\mathbf{x})$, which falls into the method of *spectral derivative*.

As an example demonstrating the use of Eq.25 to calculate the potential $\varphi(\mathbf{x})$, we set $f(\mathbf{x}) = -\frac{q}{\varepsilon}\delta(\mathbf{x})$, which represents a single point charge q in free space (i.e. a vacuum). The Fourier transform of $f(\mathbf{x})$ is:

$$\hat{f}(\mathbf{k}) = \int_{\mathbb{R}^3} [-\frac{q}{\varepsilon}\delta(\mathbf{x})] e^{-i\mathbf{kx}} d^3\mathbf{x} = -\frac{q}{\varepsilon} \tag{26}$$

as the Fourier transform of the delta function $\delta(\mathbf{x})$ is 1 over all space.

Substituting $\hat{f}(\mathbf{k})$ into Eq.25, we obtain:

$$\varphi(\mathbf{x}) = -\frac{q}{(2\pi)^3\varepsilon} \int \frac{1}{k^2} e^{i\mathbf{kx}} d^3\mathbf{k} \tag{27}$$

We shall focus on evaluating this term $\int \frac{1}{k^2} e^{i\mathbf{kx}} d^3\mathbf{k}$. It is more convenient to switch to the spherical coordinates with $d^{\mathbf{k}} = k^2 sin\theta dk d\theta d\phi$. As such, we have:

$$\int \frac{e^{i\mathbf{kx}}}{k^2} d^3\mathbf{k} = \int_0^{+\infty} \int_0^{\pi} \int_0^{2\pi} \frac{e^{i\mathbf{k}rcos\theta}}{k^2} k^2 sin\theta d\mathbf{k} d\theta d\phi \tag{28}$$

where $r = |\mathbf{x}|$ is the radius. We re-arrange the integration sequence in Eq.28 as:

$$\int \frac{e^{i\mathbf{kx}}}{k^2} d^3\mathbf{k} = \int_0^{+\infty} \int_0^{2\pi} d\phi \int_0^{\pi} e^{i\mathbf{k}rcos\theta} sin\theta d\theta d\mathbf{k} = 2\pi \int_0^{+\infty} \frac{2sin(\mathbf{k}r)}{\mathbf{k}r} d\mathbf{k} = \frac{2\pi^2}{r} \tag{29}$$

where we have made use of the integral of *sinc* function over a real line: $\int_{-\infty}^{+\infty} sinc(x)dx = \pi$. Finally, substituting this result into Eq.27 gives:

$$\varphi(\mathbf{x}) = -\frac{q}{(2\pi)^3\varepsilon} \frac{2\pi^2}{r} = -\frac{q}{4\pi\varepsilon r} \tag{30}$$

which gives the electric field $\mathbf{E}(r) = -\nabla_r\varphi(r) = \frac{q}{4\pi\varepsilon r^2}$, and the resulting force $\mathbf{F}(r) = q'\mathbf{E}(r) = \frac{qq'}{4\pi\varepsilon r^2}$, which is consistent with the result derived later by integration (Eq.52).

C.4 Solution to the d-dimensional Poisson's equation

Here we look at the generalised Poisson's equation in d-dimensional space. We can repeat the direct and inverse Fourier transforms process as in the 3D case, which gives the solution to the d-dimensional, generalised Poisson's equation:

$$\varphi(\mathbf{x}) = \frac{1}{(2\pi)^d} \int \hat{f}(\mathbf{k}) \frac{e^{i\mathbf{kx}}}{k^2} d^d\mathbf{k} \tag{31}$$

where $\mathbf{k}$ is a d-dimensional wavenumber, $k^2 = \sum_{i=1}^d k_{x_i}^2$, and $d^d\mathbf{k} = dk_{x_1} dk_{x_2} ... dk_{x_d}$.

We can also obtain the derivative:

$$\nabla_{\mathbf{x}}\varphi(\mathbf{x}) = \frac{1}{(2\pi)^d} \int (i\mathbf{k}) \hat{f}(\mathbf{k}) \frac{e^{i\mathbf{kx}}}{k^2} d^d\mathbf{k} \tag{32}$$

The problem with Eq.31 (as well as Eq.32) is the intractability of the inverse Fourier integration (which involves multiple integral in spherical coordinates), particularly when $\hat{f}(\mathbf{k})$ is complex. Here we take another approach, i.e. using direct integration to solve the Poisson's equation in a general d-dimensional space with a point charge. Similar to the 3D example, a single point charge at $\mathbf{x}$ can be

represented by $\rho(\mathbf{x}) = q\delta(\mathbf{x})$, where q is the total charge. The Poisson's equation for electrostatics (Eq.16), correspondingly, can be written as:

$$\Delta\varphi = \nabla^2\varphi = -\frac{q}{\varepsilon}\delta(\mathbf{x}) \tag{33}$$

the solution to this single point charge Poisson's equation is the potential field $\varphi(\mathbf{x})$, and therefore the electric field $E(\mathbf{x})$ as per their relation Eq.14 (and further the force), caused by the given electric charge $q\delta(\mathbf{x})$.

In Cartesian coordinates $(x_1, x_2, ..., x_d)$, Eq.33 becomes:

$$\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}\varphi(\mathbf{x}) = -\frac{q}{\varepsilon}\delta(\mathbf{x}) \tag{34}$$

Due to the problem's spherical symmetry in d dimensions, it is natural to switch to hyper-spherical coordinates. The Laplacian for a radially symmetric function $\varphi(r)$ in hyper-spherical coordinates reduces to:

$$\mathcal{L}[\varphi(r)] = \frac{1}{r^{d-1}}\frac{d}{dr}(r^{d-1}\frac{d\varphi}{dr}) \tag{35}$$

substituting the Laplacian $\mathcal{L}(\varphi(r)$ into the RHS of the Poisson's equation (Eq.34) in in hyper-spherical coordinates, we have:

$$\frac{1}{r^{d-1}}\frac{d}{dr}(r^{d-1}\frac{d\varphi}{dr}) = -\frac{q}{\varepsilon}\delta(r) \tag{36}$$

Away from the origin (i.e. $r \neq 0$), we have $\delta(r) = 0$, which simplifies Eq.36 to:

$$\frac{d}{dr}(r^{d-1}\frac{d\varphi}{dr}) = 0 \tag{37}$$

integrating it gives:

$$r^{d-1}\frac{d\varphi}{dr} = C \tag{38}$$

where C is an integration constant. Solving for φ, we get:

$$\varphi(r) = \frac{C}{(2-d)r^{d-2}} + D \tag{39}$$

again D is another integration constant. We set $D = 0$, assuming the potential vanishes at infinity. To determine the constant C, we can integrate Eq.33. First, we integrate the LHS and use the divergence theorem [12]:

$$\int_V \nabla \cdot (\nabla\varphi)dV = \int_S \nabla\varphi\hat{r} \cdot dS \tag{40}$$

which converts the volume integral into a surface integral, with S being the closed surface surrounding the volume V and $\hat{r}$ a unit vector pointing outward normal to S. The quantity on the RHS, i.e. $\int_S \nabla\varphi\hat{r} \cdot dS = \int_S \mathbf{E}\hat{r}dS$ is the electric flux over the surface S (the negative sign from Eq.14 has been indicated by the outward pointing unit vector). Due to the spherical symmetry, the integral in Eq.40 is the product of the surface area of the $d-1$ dimensional hypersphere [13] and $\nabla\varphi$, i.e.

$$\int_S \nabla\varphi\hat{r} \cdot dS = S_{d-1}r^{d-1} \times \nabla\varphi \tag{41}$$

[12]The divergence theorem states that the surface integral of a vector field over a closed surface ('the flux through the surface) equals the volume integral of the divergence over the region enclosed by the surface, i.e. $\int_V \nabla \cdot F dV = \int_S F \cdot \hat{n}dS$, where S is the closed surface surrounding the volume V, and $\hat{n}$ is a unit vector directed along the outward normal to S. It converts between the surface integral and the volume integral, and generalises to arbitrary dimension. Physically, it means the sum of all sources in a region gives the net flux out of the region.

[13]A sphere in a d-dimensional space, enclosing a d-dimensional ball (called d-ball, the d-1-sphere is the boundary of the d-ball), has dimension d-1 because it has d-1 degrees of freedom due to the fixed radius. For example, the sphere defined by $x_1^2 + x_2^2 + x_3^2 + x_4^2 + = r^2$ in 4-dimensional space is called a 3-sphere (S_3).

in which S_{d-1} is the the surface area of the $d-1$ dimensional *unit* hypersphere, and the surface area of the $d-1$-sphere with radius r is scaled by r^{d-1}. S_{d-1} can be obtained [14] by using the trick of a d-dimensional Gaussian integral, which relates this surface area to a known quantity. We know the following:

$$\int_{-\infty}^{+\infty} e^{-\mathbf{x}^2} d^d x = [\int_{-\infty}^{+\infty} e^{-x^2} dx]^d = \pi^{d/2} \tag{42}$$

where we have made use of the known Euler–Poisson integral $\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}$. This integral can also be related to the surface area of the $d-1$ dimensional *unit* sphere. Writing the LHS of Eq.42 in spherical coordinates:

$$\int_{-\infty}^{+\infty} e^{-\mathbf{x}^2} d^d x = S_{d-1} \int_{0}^{+\infty} r^{d-1} e^{-r^2} dr \tag{43}$$

the integral $\int_0^{+\infty} r^{d-1} e^{-r^2} dr$ can be evaluated [15] using change of variable, i.e. substituting $u = r^2$ and $du = 2rdr$, $dr = du/2r = du/(2\sqrt{u})$ into it, which yields $\int_0^{+\infty} r^{d-1} e^{-r^2} dr = \int_0^{+\infty} (\sqrt{u})^{d-1} e^{-u} \frac{1}{2\sqrt{u}} du = \frac{1}{2} \int_0^{+\infty} u^{\frac{d}{2}-1} e^{-u} du$, which is in the form of the Gamma function $\Gamma(\frac{d}{2})$. Therefore, we have Eq.43 reduces to:

$$\int_{-\infty}^{+\infty} e^{-\mathbf{x}^2} d^d x = \frac{1}{2} S_{d-1} \Gamma(\frac{d}{2}) \tag{44}$$

Equating Eq.42 and Eq.44, we arrive at:

$$\frac{1}{2} S_{d-1} \Gamma(\frac{d}{2}) = \pi^{d/2} \tag{45}$$

which gives:

$$S_{d-1} = \frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})} \tag{46}$$

Now we go back to Eq.41. We have now $\int_S \nabla\varphi \hat{r} \cdot dS = S_{d-1} r^{d-1} \times \nabla\varphi = \frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})} \times \varphi' = \frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})} r^{d-1} \times C r^{1-d} = \frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})} \times C$ as per Eq.38 (or Eq.39). The integration of the RHS of Eq.33 is just $-q/\varepsilon$ due to the delta function. Therefore, we have the integration constant $C = -q\Gamma(\frac{d}{2})/(2\pi^{d/2}\varepsilon)$. Finally, the solution to the single point charge Poisson's equation (i.e. Eq.39) becomes:

$$\varphi(r) = \frac{-q\Gamma(\frac{d}{2})/(2\pi^{d/2}\varepsilon)}{(2-d)r^{d-2}} = \frac{q\Gamma(\frac{d}{2})}{2(d-2)\pi^{d/2}\varepsilon r^{d-2}} \tag{47}$$

After obtaining the potential φ, we can get the electric field $\mathbf{E}$ as per Eq.14

$$\mathbf{E}_q = -\frac{d\varphi}{dr} = \frac{q\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon r^{d-1}} \tag{48}$$

and the electric force [16] exerted on another charge q', by the single point charge q at a distance r from the point charge q:

$$\mathbf{F}_{q \to q'} = q' \mathbf{E}_q = \frac{q q' \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon r^{d-1}} \tag{49}$$

[14] Alternatively, one can derive the surface area by noting that, the volume of a d-ball with radius r is: $V_d(r) = r^d V_d(1) = r^d \frac{\pi^{d/2}}{\Gamma(d/2+1)}$ (one can simply verify this in 2D and 3D. NB: if r is fixed, its volume converges to zero as dimension increases, i.e. the volume of a high-dimensional unit ball is concentrated near its surface due to curse of dimensionality), then the surface area of the d-ball can be derived as the derivative of the volume w.r.t. the radius r: $S_{d-1}(r) = \frac{d}{dr} V_d(r) = d r^{d-1} \frac{\pi^{d/2}}{\Gamma(d/2+1)}$. Therefore, the surface area of the d-ball with unit radius is: $S_{d-1} = d \frac{\pi^{d/2}}{\Gamma(d/2+1)} = \frac{2\pi^{d/2}}{\Gamma(d/2)}$, which is the same as Eq.46.

[15] Note that, as we are evaluating a definite integral, when $d-1$ is an odd integer, this integral is zero. We can relax this by replacing it with indefinite integral, which gives the same result.

[16] If we are only concerning about the electric force, there is no need to calculate the potential - all we need is the potential derivative $\nabla_{\mathbf{x}} \varphi(\mathbf{x})$.

As we are concerning free space, $\varepsilon = \varepsilon_0$ which is the vacuum permittivity.

Note that, for notational simplicity, we have dropped all direction notations in our electric potential (Eq.47), field (Eq.48), force (Eq.49) expressions and their derivations, and focus on their magnitudes. The mindful can add a *unit vector* $\mathbf{e}_{q,q'}$, which points away from a positive charge and points to a negative charge, to these formula to indicate the directions of these quantities. In a Cartesian coordinate, for example, if q is at position $\mathbf{x}_q$, and q' at position $\mathbf{x}_{q'}$, the directional unit vector is then $\mathbf{e}_{q\to q'} := (\mathbf{x}_{q'} - \mathbf{x}_q)/\|\mathbf{x}_{q'} - \mathbf{x}_q)\|_d = (\mathbf{x}_{q'} - \mathbf{x}_q))/r_{q,q'}$ where $r_{i,j} = \|\mathbf{x}_{q'} - \mathbf{x}_q)\|_d$, and $\|\cdot\|_d$ is the distance measure in d-dimensional space. By defining the direction of $\mathbf{e}_{q\to q'}$, we have by default assumed repulsive to be positive. Substituting the expression of r into e.g. the force formula gives:

$$\begin{aligned}\mathbf{F}_{q\to q'}(\mathbf{x}_i, \mathbf{x}_j) &= \frac{qq'\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon r^{d-1}}\mathbf{e}_{q\to q'} = \frac{qq'\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon r^{d-1}}\frac{\mathbf{x}_{q'} - \mathbf{x}_q}{r} \\ &= \frac{qq'\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon}\frac{\mathbf{x}_{q'} - \mathbf{x}_q}{r^d} = \frac{qq'\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon}\frac{\mathbf{x}_{q'} - \mathbf{x}_q}{\|\mathbf{x}_{q'} - \mathbf{x}_q\|_2^{d/2}}\end{aligned} \tag{49b}$$

where $\|\cdot\|_2$ denotes the L_2 norm. A 2D diagram for illustrating the repulsive force and the distance is presented Fig.6.

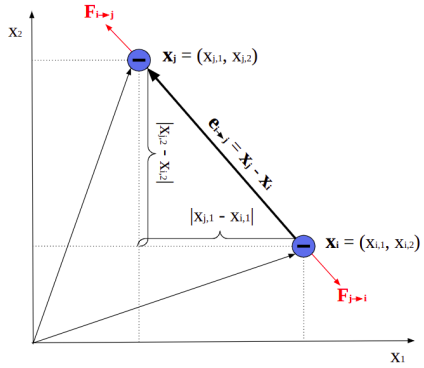


Figure 6: Diagram of repulsive force and distance in a 2D Cartesion coordinate.

Eq.49 extends the Coulomb's law for any dimensions. For example, in 1D we have:

$$\mathbf{F}_{q,q'} = \frac{qq'\sqrt{\pi}}{2\pi^{1/2}\varepsilon_0 r^{d-1}} = \frac{qq'}{2\varepsilon_0} \tag{50}$$

and for 2D:

$$\mathbf{F}_{q,q'} = q'\mathbf{E}_q = \frac{qq'}{2\pi\varepsilon_0 r} \tag{51}$$

and for 3D:

$$\mathbf{F}_{q,q'} = q'\mathbf{E}_q = \frac{qq'\sqrt{\pi}/2}{2\pi^{3/2}\varepsilon_0 r^2} = \frac{qq'}{4\pi\varepsilon_0 r^2} \tag{52}$$

which is the same as Eq.5 and Eq.30 given:

$$k = \frac{1}{4\pi\varepsilon_0} \tag{53}$$

D Repulsion and Attraction

This section provides more details about the derivation of repulsive and attracting forces.

Repulsion To mathematically formalise this idea, we first define the set of charges (interchangeably used with *particles*) $\{q_i | i = 1, 2, ..., M^{neg}\}$, where M^{neg} it the cardinality of this set, in d-dimensional space, and each is associated with position $\mathbf{x}_i$. Consider a single point charge q_i, let $\mathbf{E}_{q_i}$ be the electric field induced by q_i. In electrostatics, a electric field $\mathbf{E}$ is proportional to the electric

displacement field and equals the negative gradient of the electric potential φ; the divergence of $\mathbf{E}$, which implies the infinitesimal electric flux, is linked to the distribution of a source via the *Poisson's equation* for electrostatics (see Eq.10, Eq.14, Eq.15 in Appendix.C). We assume no external source or sink in this charge particle system, which simplifies the Poisson's equation to *Laplace's equation*; solving Laplace's equation gives the potential field φ, and therefore the electric field $\mathbf{E}$ and force $\mathbf{F}$ induced by a point charge (see Appendix.C for details). In d-dimensional space, the electric field $\mathbf{E}_{q_i}(r)$, at a distance of r from a inducing point charge q_i , is derived as:

$$\mathbf{E}_i(r) = \frac{q_i \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0 r^{d-1}}\mathbf{e} \tag{cc.Eq48}$$

in which r is the distance from the point charge q_i, ε_0 is the the vacuum permittivity constant, $\Gamma(\cdot)$ the *Gamma function*. $\mathbf{e}$ is an *unit vector* indicating the direction of the field.

Imagine we have another negative, point charge q_j locating at a distance of $r_{i,j}$ from q_i. To distinguish, we call q_i the *source point* and q_j the *field point*. The electric force exerted by q_i on q_j, as per Coulomb's law, is:

$$\mathbf{F}^{rep}_{i\to j}(r_{i,j}) = q_j \mathbf{E}_i(r_{i,j}) = \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0 r_{i,j}^{d-1}}\mathbf{e}_{i\to j} \tag{cc.Eq49}$$

where the unit vector $\mathbf{e}_{i\to j} = (\mathbf{x}_j - \mathbf{x}_i)/r_{i,j}$ gives the out-pointing direction. The arrow sign above implies 'exerting effect on', read from the left object to the right object. According to *Newton's third law*, if we treat q_j as a source point and q_i a field point, q_i experiences the same amount of force with reverse direction $\mathbf{e}_{j\to i} = -\mathbf{e}_{i\to j}$.

Now we look at the particle set $\{q_i | i = 1, 2, ..., M^{neg}\}$, and consider the combined repulsive force that particle j receives from all other particles, which can be calculated as:

$$\mathbf{F}^{rep}_j(\mathbf{x}_j) = \sum_{i=1, i\neq j}^{M^{neg}} \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0 r_{i,j}^{d-1}}\mathbf{e}_{i\to j} = \sum_{i=1, i\neq j}^{M^{neg}} \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0} \frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|_2^{d/2}} \tag{Eq49b}$$

where $\|\cdot\|_2$ denotes the L_2-norm of a vector. When charges (particles) achieve steady-state with uniform distribution, for each particle $j \in \{1, 2, ..., M^{neg}\}$, the combined repulsive force $\mathbf{F}^{rep}_j(\mathbf{x}_j)$ becomes zero, as all induced electric fields acting on the particle cancel.

One can conveniently re-write the first half of Eq.Eq49b in matrix form, by denoting the row vector $\mathbf{q}_j = [q_j, q_j, \cdots, q_j]^T$, column vector $\mathbf{q}_{1:M^{neg}} = [q_1, q_2, \cdots, q_{M^{neg}}]$, diagonal matrix $\mathbf{\Lambda} = diag[1/r_{1,j}^{d-1}, 1/r_{2,j}^{d-1}, ..., 1/r_{i,j}^{d-1}]$, as follows:

$$\mathbf{F}^{rep}_j(\mathbf{x}_j) = \frac{\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0}\mathbf{q}_j \mathbf{\Lambda} \mathbf{q}_{1:M^{neg}} \tag{Eq49c}$$

in which $\mathbf{\Lambda}$ can be viewed as some kind of isotropic covariance matrix produced by a kernel (similar to a RBF kernel but not exactly), with each diagonal element measuring a pairwise correlation or inverse-distance between the two point charges q_j and q_i.

Attraction We have devised the repulsive forces between particles of the same sign (specifically, all negative in our settings), which, under arbitrary initial positions, should generate a uniform distribution of the particles over the free space at equilibrium. To produce an uneven target distribution $p(\mathbf{x})$, we need to pull particles towards (neighbourhood) high density region using attracting forces. This can be achieved by introducing a continuous field of positive charges whose magnitudes distribute as, or proportional to, the target density. The induced new electric fields shall attract negative charges towards high probability (indicated by large positive charge magnitudes) regions. Representing a continuous density over the space requires an infinite number of positive charges to be allocated; realistically, we can mesh the space into equal-distance grids, and place large M^{pos} positive charges fixed at these grid points. As one can imagine, the number of positive charges, therefore the mesh granularity, shall impact the smoothness of the inferred density.

As such, the overall attracting force, acting on a negative charge q_j, is designed as Schmaltz et al. (2010):

$$\mathbf{F}^{attr}_j(\mathbf{x}_j) = -\sum_{i'=1, i'\neq j}^{M^{pos}} \frac{qp(\mathbf{x}_{i'}) q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0 r_{i',j}^{d-1}}\mathbf{e}_{i'\to j} = -\sum_{i'=1, i'\neq j}^{M^{pos}} \frac{qp(\mathbf{x}_{i'}) q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0} \frac{\mathbf{x}_j - \mathbf{x}_{i'}}{\|\mathbf{x}_j - \mathbf{x}_{i'}\|_2^{d/2}} \tag{Eq49d}$$

where we have set the positive charge $q_j = qp(\mathbf{x}_j)$ with q being a constant scaling factor specifying the relative strength of attracting forces. A negative sign is put ahead as it's attracting force.

If we ignore the contributing forces from those located far away, in regions with constant probability (and therefore uniform positive charge magnitudes), the attracting forces cancel and repulsion still yields the energetically ideal positions which maximize pairwise distances. In regions with uneven probability distribution, attracting force dominates and binds particles onto locations with higher probability. If mesh grid is set to be infinitely small, M^{pos} grows to infinitely large and the distribution of positive charge magnitude becomes continuous, Eq.Eq49d then becomes an integral.

The combined force Summation of the repulsive and attracting forces gives the overall force $\mathbf{F}_j = \mathbf{F}_j^{rep} + \mathbf{F}_j^{attr}$, acting on a single free-moving negative charge j:

$$\mathbf{F}_j^t = \Big[\sum_{i=1, i\neq j}^{M^{neg}} \frac{q_i q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0} \frac{\mathbf{x}_j - \mathbf{x}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|_2^{d/2}} - \sum_{i'=1, i'\neq j}^{M^{pos}} \frac{qp(\mathbf{x}_{i'}) q_j \Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0} \frac{\mathbf{x}_j - \mathbf{x}_{i'}}{\|\mathbf{x}_j - \mathbf{x}_{i'}\|_2^{d/2}} \Big] \tag{54}$$

E A NOTE ON COMPLEXITY

Most computational efforts lie in computing the interacting force between particles. At each iteration, in order to calculate the two forces for a single negative particle j, we have to calculate all the pairwise distances $1/r_{i,j}^{d-1}$ and $1/r_{i',j}^{d-1}$ where $i \in \{1, 2, ..., M^{neg}\}$ and $i' \in \{1, 2, ..., M^{pos}\}$, as shown in Figure.7. That is, we have to evaluate the following distance matrix in each iteration:

$$R_{(M^{neg}+M^{pos})\times(M^{neg}+M^{pos})} = \begin{array}{c} \\ q_1 \\ q_2 \\ \cdots \\ q_{M^{neg}} \\ \cdots \\ q_{M^{pos}} \end{array} \begin{array}{c} \begin{array}{cccccc} q_1 & q_2 & \cdots & q_{M^{neg}} & \cdots & q_{M^{pos}} \end{array} \\ \begin{pmatrix} 0 & r_{1,2} & \cdots & r_{1,M^{neg}} & \cdots & r_{1,M^{pos}} \\ r_{2,1} & 0 & \cdots & r_{2,M^{neg}} & \cdots & r_{2,M^{pos}} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ r_{M^{neg},1} & r_{M^{neg},2} & \cdots & 0 & \cdots & r_{M^{neg},M^{pos}} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ r_{M^{pos},1} & r_{M^{pos},2} & \cdots & r_{M^{pos},M^{neg}} & \cdots & 0 \end{pmatrix} \end{array} \tag{55}$$

which can be pre-calculated or calculated when needed. Calculating the whole matrix R takes order $\mathcal{O}((M^{neg} + M^{pos})^2)$. The computational efforts will be huge if either M^{neg} or M^{pos} is large. Of course, half of the computation can be saved by making use of the symmetry $r_{i,j} = r_{j,i}$.

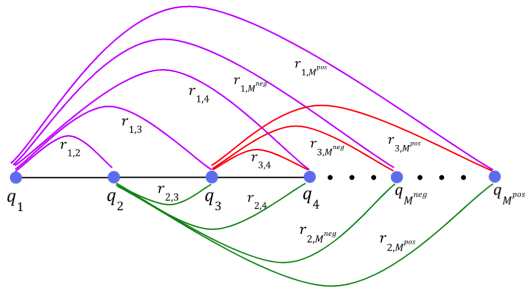


Figure 7: Pairwise distances.

Let's take a closer look at the repulsive force in Eq.Eq49c by expanding it in matrix form:

$$\begin{aligned}\mathbf{F}_j^{rep}(\mathbf{x}_j) &= \frac{\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0}\mathbf{q}_j\mathbf{\Lambda}\mathbf{q}_{1:M^{neg}} \\ &= \frac{\Gamma(\frac{d}{2})}{2\pi^{d/2}\varepsilon_0}\begin{pmatrix} q_j & q_j & \cdots & q_j \end{pmatrix}\begin{pmatrix} 1/r_{1,j}^{d-1} & 0 & \cdots & 0 \\ 0 & 1/r_{2,j}^{d-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/r_{i,j}^{d-1} \end{pmatrix}\begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_{M^{neg}} \end{pmatrix}\end{aligned} \tag{cc.Eq49c}$$

similar matrix expansion can be written with the attracting force in Eq.Eq49d for particle j.

When calculating $\mathbf{F}_j^{rep}(\mathbf{x}_j)$ and $\mathbf{F}_j^{attr}(\mathbf{x}_j)$ for the particle j, we need to retrieve the relevant pairwise distance vector, i.e. the row $r_{j,\cdot}$ or column $r_{\cdot,j}$, from the distance matrix R in Eq.55. However, if necessary (i.e. if either M^{neg} or M^{pos} is large), we can take a neighbourhood search to identify a neighbourhood set and calculate only the pairwise distances between particles within the neighbourhood set. This shrinks the distance vector for particle j.

F TOY EXAMPLES: MORE DETAILS

Gaussian densities The two target Gaussian densities are as follows:
Unimodal:

$$p(\mathbf{x}) = \mathcal{N}(\mu, \Sigma)$$

with $\mu = (0.5, 0.5)$ and $\Sigma = 0.05I$.
Bimodal:

$$p(\mathbf{x}) = 0.7\mathcal{N}(\mu_1, \Sigma_1) + 0.3\mathcal{N}(\mu_2, \Sigma_2)$$

with $\mu_1 = (0, 0)$, $\Sigma_1 = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$, $\mu_2 = (4, 4)$, $\Sigma_2 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$.

In inferring the unimodal Gaussian, we draw negative charges from the uniform prior: $p^0(\mathbf{x}) = U(0, 0.5) \times U(0, 0.5)$; In inferring the bimodal Gaussian, we draw negative charges from the uniform prior: $p^0(\mathbf{x}) = U(-3, 7) \times U(-3, 7)$.

Other three densities The other 4 un-normalised target densities used are:
Moon-shared Haario et al. (1999); Wang et al. (2021):

$$\rho(x) \propto \exp\left\{-\frac{x_1^2}{2} - \frac{1}{2}\left(10x_2 + 3x_1^2 - 3\right)^2\right\}$$

Double-banana Rezende & Mohamed (2016b); Liu et al. (2019a):

$$\rho(x) \propto \exp\left\{-2\left((x_1^2 + x_2^2 - 3)^2\right) + \log\left(e^{-2(x_1-2)^2} + e^{-2(x_2+2)^2}\right)\right\}$$

Wave density Rezende & Mohamed (2016b); Chen et al. (2018); Wang et al. (2021):

$$\rho(x) \propto \exp\left\{-\frac{1}{2}\left[\frac{x_2 - \sin\left(\frac{\pi x_1}{2}\right)}{0.4}\right]^2\right\}$$

Implementation The mesh grid for all 3 other densities is within $[-3, 3] \times [-3, 3]$; for the bi-modal Gaussian it is $[-3, 7] \times [-3, 7]$; for uni-modal Gaussian it is $[0, 1] \times [0, 1]$. The 400 negative charges are initialised with 2D standard Gaussian for the moon-shaped case, and uniformly for all other cases (uniform within the whole mesh grid, except for the uni-modal Gaussian case where the initialisation area is $[0, 0.5] \times [0, 0.5]$).

Euler updating rule (Eq.1) is used and $M^{pos} = 50 \times 50$, $M^{neg} = 400$, $\sigma_{noise} = 0$ applied. The overall forces across negative charges are normalised in each iteration before calculating the displacement. The charge magnitudes $q^{neg} = q = 1.0$. Permittivity of free space $\varepsilon_0 = 8.854e - 12$. Euler proportionality constant $\tau = 0.1$. Particles are evolved for $T = 100$ iterations. For all three densities, at equilibrium, we discard the particles outside the grid area.

G NEAL'S FUNNEL: IMPLEMENTATION DETAILS

The MMD^2 metric is defined as Arbel et al. (2019); Wang et al. (2021):

$$\text{MMD}^2 = \frac{1}{N^2}\sum_{i,j=1}^{N} k(x_i, x_j) + \frac{1}{M^2}\sum_{i,j=1}^{M} k(y_i, y_j) - \frac{2}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} k(x_i, y_j) \tag{56}$$

with the polynomial kernel $k(x, y) = \left(\frac{x^\top y}{3} + 1\right)^3$.

HMC The *PyMC* Python API Oriol et al. (2023) is used to implement the HMC method. In total, 1 chain with 4000 samples is drawn, with 1000 burn-up samples discarded. These 4000 samples are then thinned evenly to produce 400 samples.

MH Initialised from near zero, we use a standard normal distribution as the proposal distribution, the simplified acceptance probability is defined as the ratio of the new sample to current position.

LMC We initialise 400 samples uniformly within $[-7, 3] \times [-7, 3]$, and evolve them through 10000 iterations; in each iteration, each particle is evolved through the *Langevin dynamics* Wang et al. (2019):

$$\begin{aligned}
\mathbf{x}_{t+1} &= \mathbf{x}_t + \epsilon_t \nabla \log p(\mathbf{x}_t) + \sqrt{2\epsilon_t}\mathbf{z}_t \\
\epsilon_t &= a(b+t)^{-c} \\
\mathbf{z}_t &\sim \mathcal{N}(0, I)
\end{aligned}$$

with $a = 0.01, b = 1, c = 0.55$ and 10000 iterations for each sample. t is the iteration number.

We also records the step time, mean negative log-likelihood (NLL), and squared maximum mean discrepancy squared (MMD^2) in each iteration.

SVGD SVGD inference updates particle positions as per Liu & Wang (2016):

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \epsilon \phi^*_{q,p}(\mathbf{x}_i^l)$$

and

$$\phi^*(\mathbf{x}_i^l) = \frac{1}{M}\sum_{j=1}^{M}[k(\mathbf{x}_i^l, \mathbf{x}_j^l)\nabla_{\mathbf{x}} \log p(\mathbf{x}_j^l) + \nabla_{\mathbf{x}_j} k(\mathbf{x}_i^l, \mathbf{x}_j^l)]$$

We referred to the implementation by Wang et al. (2021), in which a RBF kernel $\exp(\|x - x'\|/2h^2)$ is used with the hyper-parameter length-scale h=median pairwise distance. The step size used is 0.01. The same 400 uniformly initialised samples (as used in LMC) are evolved, with total number of iterations 100,000. We also records the step time, mean NLL, and MMD^2 in each iteration.

EVI We refer to the implementation by Wang et al. (2021), in which the RBF kernel with length-scale $h = 0.1$ is used. The step size used is 1.0×10^-7. The same 400 uniformly initialised samples (as used in LMC) are evolved, with total number of iterations 2,500,000. We also records the step time, mean NLL, and MMD^2 in each iteration.

EParVI The 400 negative charges are initialised uniformly, the same as that of LMC, SVGD and EVI. Euler updating rule (Eq.1) is used and $M^{pos} = 10,000, M^{neg} = 400, \sigma_{noise} = 0$ applied. The overall forces across negative charges are normalised in each iteration before calculating the displacement. The charge magnitudes $q^{neg} = q = 1.0$. Permittivity of free space $\varepsilon_0 = 8.854e - 12$. Euler proportionality constant $\tau = 0.1$. Particles are evolved for $T = 100$ iterations. The MMD^2 and NLL metrics are evaluated based on particles within the mesh grid.

H BAYESIAN LOGISTIC REGRESSION: MORE DETAILS

Posterior The likelihood can be written as:

$$p(\boldsymbol{y}|\mathbf{x},\boldsymbol{\omega}) = \prod_{i=1}^{N}\left[\frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}\right]^{y_i}\left[1-\frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}\right]^{1-y_i}$$

combining with the prior:

$$p(\boldsymbol{\omega}) = \mathcal{N}(\boldsymbol{\omega};\mathbf{0},\alpha\boldsymbol{I}) = \frac{1}{(2\pi\alpha)^{d/2}}\exp\left(-\frac{1}{2\alpha}\boldsymbol{\omega}^T\boldsymbol{\omega}\right)$$

where $d = 4$ is the number of features (not considering intercept), we obtain the coefficients $\boldsymbol{\omega}$ posterior:

$$p(\boldsymbol{\omega}|\boldsymbol{y},\mathbf{x}) \propto \prod_{i=1}^{N}\left[\frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}\right]^{y_i}\left[1-\frac{1}{1+\exp(-\boldsymbol{\omega}^T\mathbf{x}_i)}\right]^{1-y_i}\cdot\exp\left(-\frac{1}{2\alpha}\boldsymbol{\omega}^T\boldsymbol{\omega}\right)$$

Taking log we have:

$$\log p(\boldsymbol{\omega}|\boldsymbol{y},\mathbf{x}) = \sum_{i=1}^{N}\left[y_i\log\sigma(\boldsymbol{\omega}^T\mathbf{x}_i) + (1-y_i)\log(1-\sigma(\boldsymbol{\omega}^T\mathbf{x}_i))\right] - \frac{1}{2\alpha}\boldsymbol{\omega}^T\boldsymbol{\omega} + \text{constant}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the logistic function. One can then conveniently take gradient w.r.t ω to use gradient-based sampling algorithms such as HMC.

Gradient of posterior The gradient of $\log\sigma(\omega^T\mathbf{x}_i)$ w.r.t. ω is:

$$\nabla_\omega \log\sigma(\omega^T\mathbf{x}_i) = (1-\sigma(\omega^T\mathbf{x}_i))\mathbf{x}_i$$

The gradient of $\log(1-\sigma(\omega^T\mathbf{x}_i))$ w.r.t. ω is:

$$\nabla_\omega \log(1-\sigma(\omega^T\mathbf{x}_i)) = -\sigma(\omega^T\mathbf{x}_i)\mathbf{x}_i$$

Combining these, the gradient of the likelihood term is:

$$\sum_{i=1}^{N}\left[y_i(1-\sigma(\omega^T\mathbf{x}_i))\mathbf{x}_i - (1-y_i)\sigma(\omega^T\mathbf{x}_i)\mathbf{x}_i\right]$$

which simplifies to:

$$\sum_{i=1}^{N}\left[(y_i-\sigma(\omega^T\mathbf{x}_i))\mathbf{x}_i\right]$$

On the other hand, the gradient of the prior term w.r.t. ω is:

$$-\frac{1}{\alpha}\omega$$

Combining the gradients of the likelihood and the prior, we get:

$$\nabla_\omega \log p(\omega|y,X) = \sum_{i=1}^{N}\left[(y_i-\sigma(\omega^T\mathbf{x}_i))\mathbf{x}_i\right] - \frac{1}{\alpha}\omega$$

In matrix form, this can be written as:

$$\nabla_\omega \log p(\omega|y,\mathbf{X}) = \mathbf{X}^T(y-\sigma(\mathbf{X}\omega)) - \frac{1}{\alpha}\omega$$

where $\sigma(\mathbf{X}\omega)$ is the vector of sigmoid values for each row of $\mathbf{X}$.

Implementation The Iris dataset contains 150 observations, each instance with 4 features. The whole dataset is 70%:30% split into training and test sets, with training contains 105 instances and test has 45 instances. Each observation is represented by the feature-label pair $(\mathbf{x}_i, y_i)$. Features are normalised before being used.

The SVGD, EVI and EParVI methods use the same number of particles (i.e. 400), and share the same initialisation. Specifically:

HMC In total, 1 chain with 2000 samples is drawn. After discarding the 1000 burn-up samples, the first 400 samples are used.

SVGD A RBF kernel $\exp(\|x - x'\|/2h^2)$ is used with the hyper-parameter length-scale h=median pairwise distance. The step size used is 0.01. Total number of iterations is 200,000.

EVI A RBF kernel with length-scale $h = 0.1$ is used. The step size used is 1×10^-7. Total number of iterations is 2.5×10^8.

EParVI Euler updating rule (Eq.1) is used and $M^{pos} = 12^4, M^{neg} = 400, \sigma_{noise} = 0$ applied. Note the region spans $[-3, 3] \times [-3, 3]$, and the granularity (i.e. resolution) is therefore 0.5. The overall forces across negative charges are normalised in each iteration before calculating the displacement. The charge magnitudes $q^{neg} = 1.0$ and $q = 1.0 \times 10^7$. The relatively large value of q is due to small chain product of the un-normalised likelihood in Eq.4; to make it on the same scale as the repulsive force, by trial and error we need to set a large value for q. a strategy to avoid small (absolute) values of the likelihood is to take log, as we did in inferring the LV dynamical system parameters in Section.I. Euler proportionality constant $\tau = 0.1$. Particles are evolved for $T = 60$ iterations.

Implementation tricks For complex geometries, more steps and more charges are required. Updating the particle positions using different rules may lead to different results. Interestingly, in our experiments we find that, if we update the negative positions according to Eq.1c, in the limit all negative charges are absorbed to high probability grid points. Also, if the Δt used in the Verlet or damped Verlet methods is too large, then the particle system is unstable and cannot converge to a steady-state.

One can add random noise to perturb the negative charge positions to avoid overlapping at grid points and to avoid getting trapped in local optima. One can choose to discard particles falling out of the grids (e.g. those get trapped in low-density regions) at the end or sequentially. This also saves computational efforts. Without an MH step, charges can take large, risky steps and jump outside the region. Hanson (2005) also addressed that, appropriate conditions need to be specified at the boundary of the region. In our experiments, we designed an annealing scheme for reducing q^{pos} over iterations to promote within-mode diversity.

It is generally suggested to pre-evaluate the relative magnitudes of repulsive and attracting forces and scale q^{pos} properly to ensure they are on the same scale; otherwise, e.g. when repulsive forces is of order larger than attracting force, the density center will never attract the particles. Effect of the ratio q/q_i is important: a small ratio can make the repulsive force dominate and lead to explosion; a large value can accelerate convergence, but also be too restrictive - the final positions of negative charges overlap and diversity reduces.

I INFERENCE OF THE LOTKA-VOLTERRA DYNAMICAL SYSTEM

Here we look at the Lotka-Volterra (LV) ecological system whose dynamics can be represented by the following two ODEs:

$$\begin{aligned} \frac{dx}{dt} &= ax - bxy \\ \frac{dy}{dt} &= cxy - dy \end{aligned} \tag{57}$$

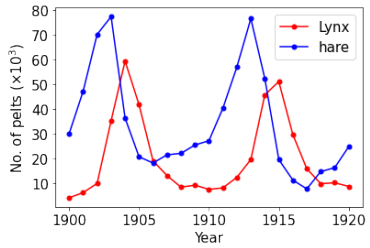


Figure 8: Temporal evolution of the LV system.

where x and y are the numbers of hare and lynx pelts. We are interested in inferring the for inferring the 4-dimensional parameter $\theta = (a, b, c, d)$ based on 21 observed real-world data as plotted in Fig.8. This can be formulated as a Bayesian inference problem with the posterior:

$$p(\theta|X', Y') = p(a, b, c, d|X', Y') = \frac{p(X', Y'|a, b, c, d)p(a)p(b)p(c)p(d)}{Z} \tag{58}$$

where Z is the normalising constant. The observed trajectories $(X', Y') = \{(x'_i, y'_i)\}_{i=1}^{21}$, and independence of parameters is assumed. The priors are specified as:

$$a, d \sim U(0.001, 1.0), \quad b, c \sim U(0.001, 0.05) \tag{59}$$

and likelihood:

$$\log x'(t_i) \sim \mathcal{N}(\log x(t_i), 0.25^2), \quad \log y'(t_i) \sim \mathcal{N}(\log y(t_i), 0.25^2) \tag{60}$$

where we have for simplicity assumed known [17], homogeneous state noise level $\sigma = 0.25$, and known initial conditions $x_0 = 33.956, y_0 = 5.933$. Assuming independence between the states writes the likelihood as:

$$\begin{aligned} p(X', Y'|a, b, c, d) &= p(X'|a, b, c, d)p(Y'|a, b, c, d) = \prod_{i=1}^{N} p(x'_i|a, b, c, d) \prod_{i=1}^{N} p(y'_i|a, b, c, d) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \frac{1}{x'_i} exp[-\frac{(\log x'_i - \log x_i)^2}{2\sigma^2}] \frac{1}{\sqrt{2\pi}\sigma} \frac{1}{y'_i} exp[-\frac{(\log y'_i - \log y_i)^2}{2\sigma^2}] \end{aligned} \tag{61}$$

where $x(t_i)$ and $y(t_i)$ are the solutions to Eq.57, $x'(t_i)$ and $y'(t_i)$ are the observed data in Fig.8.

Substituting the priors (Eq.59) and likelihood (Eq.61) into the posterior (Eq.58), we obtain the probabilistic model $p(\theta|X', Y')$, which is the target of our inference practice. Normally, logarithm of the posterior is preferred. The logarithm of the posterior (Eq.58), with priors written in Eq.59 and likelihood in Eq.61, is:

$$\log p(\theta|X', Y') = \log p(a, b, c, d|X', Y') = \log p(a) + \log p(b) + \log p(c) + \log p(d) + \log p(X', Y'|a, b, c, d) = \sum_{i=a,b,c,d} \log(\frac{1}{max_i - min_i}) - 2N \log(\sqrt{2\pi}\sigma) - \sum_{i=1}^{N} (\log x'_i + \log y'_i) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} [(\log x'_i - \log x_i)^2 + (\log y'_i - \log y_i)^2] \tag{62}$$

[17] These known values are referenced from Carpenter (2022)

where $min_a = min_b = min_c = min_d = 0.001, max_a = max_b = max_c = max_d = 1$ as specified by Eq.59, and noise level $\sigma = 0.25$. Note that, though not explicitly shown, the ODE solutions (x_i, y_i) are functions of $\theta = (a, b, c, d)$.

As the score $\nabla_\theta \log p(\theta|X', Y')$ is used in gradient-based samplers such as LMC and HMC, we derive it as follows [18]:

$$\begin{aligned}
\frac{\partial \log p(a,b,c,d|X',Y')}{\partial a} &= \frac{1}{\sigma^2}[(\frac{\log X' - \log X}{X})^T \frac{\partial X}{\partial a} + (\frac{\log Y' - \log Y}{Y})^T \frac{\partial Y}{\partial a}] \\
\frac{\partial \log p(a,b,c,d|X',Y')}{\partial b} &= \frac{1}{\sigma^2}[(\frac{\log X' - \log X}{X})^T \frac{\partial X}{\partial b} + (\frac{\log Y' - \log Y}{Y})^T \frac{\partial Y}{\partial b}] \\
\frac{\partial \log p(a,b,c,d|X',Y')}{\partial c} &= \frac{1}{\sigma^2}[(\frac{\log X' - \log X}{X})^T \frac{\partial X}{\partial c} + (\frac{\log Y' - \log Y}{Y})^T \frac{\partial Y}{\partial c}] \\
\frac{\partial \log p(a,b,c,d|X',Y')}{\partial d} &= \frac{1}{\sigma^2}[(\frac{\log X' - \log X}{X})^T \frac{\partial X}{\partial d} + (\frac{\log Y' - \log Y}{Y})^T \frac{\partial Y}{\partial d}]
\end{aligned} \tag{63}$$

where the posterior with uniform priors in Eq.62 is used. The parameter sensitivities $(\frac{\partial X}{\partial a}, \frac{\partial X}{\partial b}, \frac{\partial X}{\partial c}, \frac{\partial X}{\partial d})$ can be solved by differentiating the LV dynamics in Eq.57.

To make all log posteriors positive, we use the following offset log posterior as target for EParVI:

$$\log p(\theta|X', Y') - \log p(\theta_0|X', Y') = \log \frac{p(\theta|X', Y')}{p(\theta_0|X', Y')}$$

where $p(\theta_0|X', Y') = \min_{\theta \in \mathcal{G}} \log p(\theta|X', Y')$, $\mathcal{G}$ is the set of all grid points (locations of positive charges). Alternatively, $p(\theta_0|X', Y')$ can be chosen to be a value smaller than the log posterior evaluated at any point in the space of consideration.

To make the learning more numerically stable [19], we set the positive charge magnitudes [20] $q_j = q \log p(\theta_j|X', Y')$ where q is a scaling factor which specifies the maximum magnitude, as shown in Eq.62.

Before applying EParVI, we do a quick check of the probabilistic model. We evaluate $p(\theta|X', Y')$ over all mesh grid points, and compare its maximum, i.e. the maximum a posterior (MAP) estimate, with reference values. The MAP estimate is

$$\hat{\theta}^{MAP} = argmax_j p(\theta_j|X', Y') \equiv argmax_j \log p(\theta_j|X', Y') = [\overset{a}{0.539}, \overset{b}{0.027}, \overset{c}{0.024}, \overset{d}{0.795}]$$

which is very close to the ground truth value from literature Carpenter (2022). This confirms that, with the current meshing granularity, the peak of the positive charge magnitude aligns with ground truth.

By querying the log posterior formula, we obtain the log posterior values at 810000 equidistant points (30 along each dimension). Marginalising out two dimensions, we arrive at the theoretical, 2D marginal log posteriors as presented in Fig.9. The whole approximate landscape of $p(\theta|X', Y')$ is visualised in Fig.10. We observe that, the log posterior surface is relatively flat, which could potentially impact the identifiability of modes.

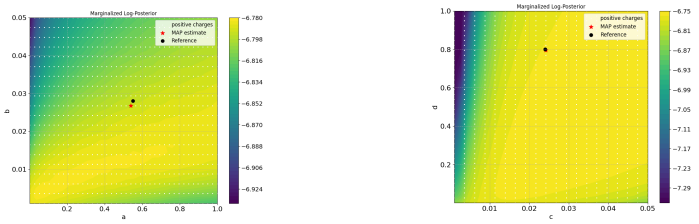
Then we run 5 methods, i.e. MAP, HMC, LMC, SVGD and EParVI, to infer the posterior $p(\theta|X', Y')$, using similar model settings [21], and compare their performances. For EParVI, we first simulate the LV

[18] More details can be found in the author's coming book "Some empirical observations on Stein variational gradient descent".

[19] The chain product in the posterior Eq.58 leads to small absolute values of the target, and thus requires large positive magnitude q.

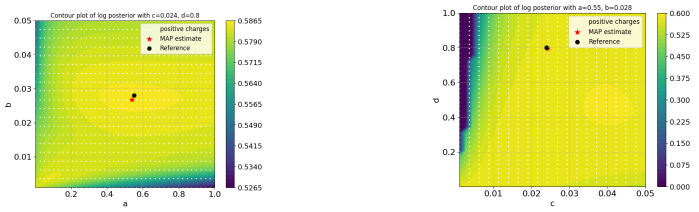
[20] Note that, it is crucial for the charge magnitudes q_i, where $i \in \{1, 2, ...M^{neg}\}$, and $q_{i'}$, where $i' \in \{1, 2, ...M^{pos}\}$, to remain positive. While we use q_i=1.0 in all experiments; the magnitudes of positive charges $q_{i'}$, however, can be negative in scenarios e.g. $q_{i'} = \log p(\mathbf{x}_{i'})$. Therefore, an offset is made, i.e. $q_{i'} = \log p(\mathbf{x}_{i'}) - min[\log p(\mathbf{x}_{i'})]$ to ensure positivity. Further, at grid points which results in negative values of $(x(t), y(t))$, we set $q_{i'}$=0.

[21] The same priors are used in all six methods; HMC, however, for convenience of implementation (mainly due to the negativity of some trajectory values at impossible θ locations, which results in error when taking log), a different likelihood is used: $x'(t_i) \sim \mathcal{N}(x(t_i), \sigma^2)$ and $y'(t_i) \sim \mathcal{N}(y(t_i), \sigma^2)$.



(a) Marginalising out c and d. (b) Marginalising out a and b.

Figure 9: Marginal log posterior distributions. The left figure marginalises out parameters c and d, while the right figure marginalises out a and b.



(a) Snapshot at $c = 0.024$, $d = 0.8$. (b) Snapshot at $a = 0.55$, $b = 0.028$.

Figure 10: Log posterior contour plots. Left: snapshot at $c = 0.024, d = 0.8$; right: snapshot at $a = 0.55, b = 0.028$.

system at a grid of mesh points (Fig.10), obtaining (X, Y) and substituting (X, Y) and (X', Y') into the log posterior to obtain the positive charge magnitudes at these mesh points. These magnitudes are cached for later query. Then we randomly initialise $\theta = (a, b, c, d)$, i.e. the coordinates of negative charges, and calculate the pairwise distances between each negative charge and all other charges, further the repulsive and attracting forces (by querying the positive charge magnitudes) acting on each negative charge. The overall force is then substituted into the updating rule to give the displacement for each negative charge. We run the EParVI algorithm till stability using 400 negative and 640000 positive particles, the initial and final configurations of the particles, along with their empirical marginal distributions, are presented in Fig.11 (more details see Fig.13 in Appendix.**??**). It is seen that, the inferred geometries for b, c are narrow and peaky, while those for a, d are flat and widely distributed; final particles sit in the high probability regions of the posterior, which is consistent with the posterior snapshots in Fig.10 - particles along the dimensions b and c are moer concentrated than in dimensions a and d. The relatively wide spread of the posterior distributions of a and d may be due to the fact that Ročková (2021), realizations of the LV system are more sensitive to the interaction parameters b and c, while less sensitive to a and b. The posterior-based predictions are presented in Fig.12, from which we observe loosely good match between the empirical posterior mode-based predictions and real trajectories, with some lifts and shifts. The predictions made using all final particles, which represent uncertainties, exhibit large variations.

(a) Empirical marginal distribution of a and b. (b) Empirical marginal distribution of c and d.

Figure 11: EPaVI: initial and final particle distributions, showing the empirical marginal distributions of key variables.

A comparison of the inference results is made in Table.2. We observe that, the EParVI estimations are close to the reference values, particularly in dimensions b and c. It, however, also occupies the

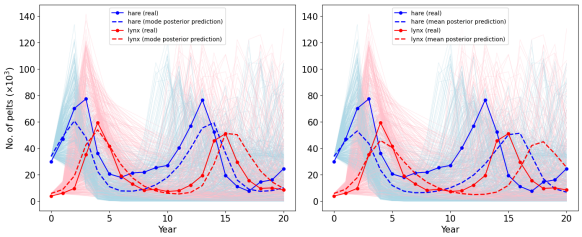


Figure 12: EParVI posterior-based trajectory predictions. Left: predictions made using mean posterior, right: predictions made using posterior mode. In both figures, light blue and pink lines represent the predictions for hare and lynx, respectively, made by all final particles.

longest runtime. EParVI outperforms SVGD and LMC; the optimal choice, balancing accuracy and efficiency, however is HMC.

Method\Quantity	a	b	c	d	runtime(s)[1]
EParVI (mean/mode)	0.487/**0.549**	0.026/0.026	**0.024**/0.025	0.529/0.645	221400
SVGD (mean/mode)	1.017/1.029	0.076/0.073	0.045/0.032	1.025/0.978	8662
LMC (mean/mode)	0.482/0.556	0.028/0.019	0.021/0.021	0.555/0.651	24178
HMC (mean/mode)	0.566/0.570	**0.028**/0.027	0.023/0.023	**0.763**/0.759	**366**
MAP (EParVI mesh)	0.539	0.027	0.024	0.795	-
Reference[2]	<u>0.55</u>	<u>0.028</u>	<u>0.024</u>	<u>0.80</u>	-

[1] These methods can differ in numbers of samples and iterations, see Appendix.**??**.

Table 2: Inference results of the LV system coefficients $\theta = [a, b, c, d]$ using six methods.

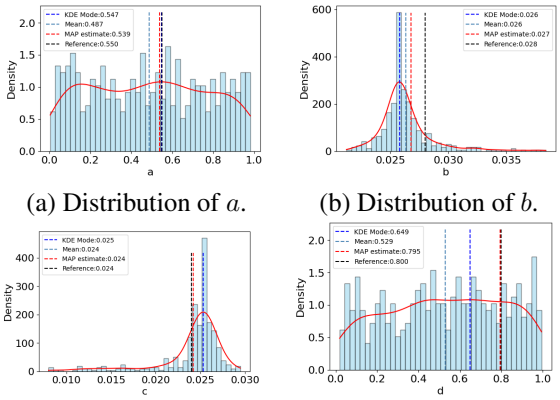
Implementation details and results Implementation details and supplementary results are listed as follows:

EParVI The uniform prior space spans $[0.001, 1] \times [0.001, 0.05] \times [0.001, 0.05] \times [0.001, 1]$. Each dimension has 40,20,20,40 grid points, respectively. Total number of positive charges $M^{pos} = 40 \times 20 \times 20 \times 40 = 640000$. Total number of negative charges $M^{neg} = 400$. Positive charge magnitudes $q_j = q \log p(\theta_j | X', Y')$ are pre-calculated before all iterations, with maximum magnitude $q = 10^{-5}$. Noise level $\sigma_{noise} = 0$. Euler proportionality constant $\tau = 0.01$. Particles are evolved for $T = 82$ iterations. Each iteration takes ~ 2700 seconds to run. Overall forces across negative charges are normalised in each iteration before Euler updates. Total runtime for 82 iterations is ~ 221400 seconds. Particle positions marginally change after 50 iterations. The empirical, marginal distribution for each dimension, calculated from the 400 particles, is pictured in Fig.13.

LMC We use similar LMC implementation as described in Appendix.G, with $a = 1e-8, b = 1, c = 0.55$. 400 $\mathbf{x}_0$ are initialised uniformly (same initial samples used in EParVI), and use Langevin dynamics to trace each sample through 10000 steps. Total runtime is 24178 seconds. The final positions of the 400 samples are presented in Fig.14. The final shapes are somewhat consistent with the theoretical snapshots in Fig.10, with much dispersion.

HMC The No-U-Turn sampler (NUTS) Homan & Gelman (2014), an advanced HMC implementation, is used with PyMC3 library Oriol et al. (2023). The author crafted the specific ODEOp and solver. Two chains, each with 1900 samples, are drawn with target acceptance rate 0.8. Total runtime is 366 seconds. The first chain is used, with 1500 burn-in samples discarded. Initial conditions used are $x_0 = 33.956, y_0 = 5.933$. Note that, the likelihood used for HMC is $x'(t_i) \sim \mathcal{N}(x(t_i), (e^{0.25})^2)$ and $y'(t_i) \sim \mathcal{N}(y(t_i), (e^{0.25})^2)$, which is different from that of EParVI specified in Eq.60.

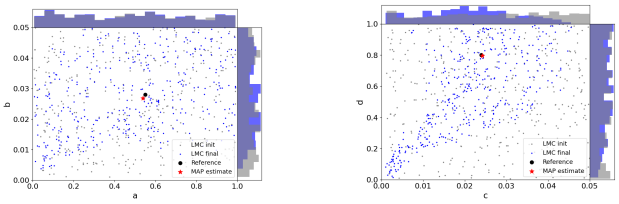
The two chains are presented in Fig.15, with left showing the sample histogram based KDE density and on the right sample trajectories. We observe that the two chains agree on the estimations, and the samples, although small numbers, concentrate around the the means and modes. We extract one



(a) Distribution of a. (b) Distribution of b.

(c) Distribution of c. (d) Distribution of d.

Figure 13: Empirical marginal distributions for a, b, c, and d.



(a) Particle distributions for a and b. (b) Particle distributions for c and d.

Figure 14: EPaVI: Initial and final particle distributions.

chain (yellow one in Fig.15) and take a closer look in Fig.16 at the sample estimates, we observe that, given our HMC model, the results are very close to the reference values (note the small horizontal scales). These empirically estimated values are summarised in the third row of Table.2. We then make predictions using the empirically estimated posterior means and MAPs based on this single chain, as shown in Fig.17. Again, we observe good match with the real world recorded data.

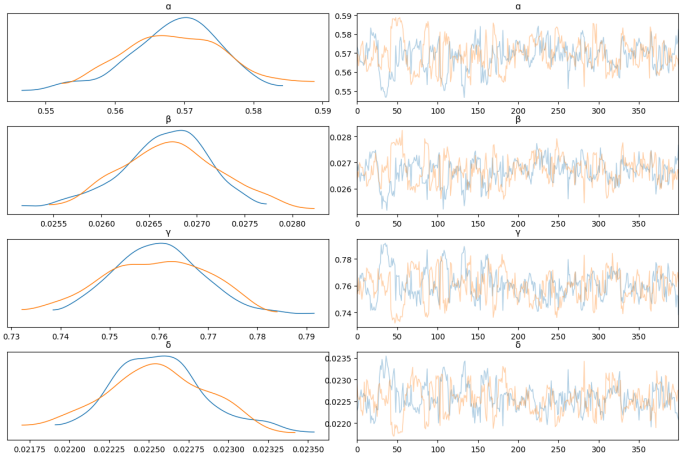
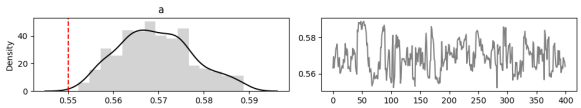
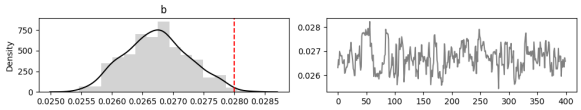


Figure 15: HMC samples. In the figure, $\alpha = a$, $\beta = b$, $\delta = c$, $\gamma = d$.

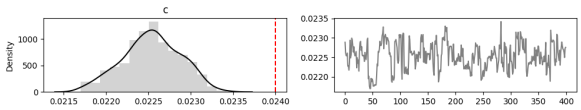
SVGD Total number of initial particles: M=2000, overall number of iterations: $L = 100$, kernel: RBF ($2h^2$=$med^2/\log M$ where med is the median distance between particles), learning rate: ϵ=$0.001/\|\phi^*(\boldsymbol{z}_i^l)\|_2$, where i is the particle index, $l = 1, 2, ..., L$ is the iteration number. Total runtime time: 8662 seconds. Valid number of particles after 100 iterations: 1454.



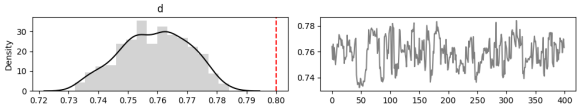
(a) Parameter a.



(b) Parameter b.



(c) Parameter c.



(d) Parameter d.

Figure 16: *Lotka-Volterra* system identification: HMC inference of $\theta = [a, b, c, d]$ using the real-world data shown in Fig. 8. The red dashed vertical line represents the reference value from Carpenter (2022).

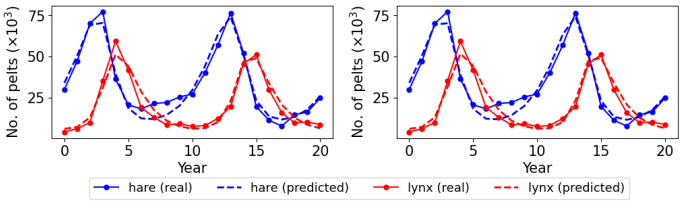


Figure 17: HMC inferred posterior predictives. Left: MAP-based predictions. Right: mean-based predictions.

EVI EVI is not used due to our intractability of negative states encountered in simulating the LV dynamics when calculating the gradients.

J Mapping concepts in electrostatics and probability

Quantities in electrostatics can be mapped to the concepts in probability distributions as follows:

Output Type	**Electrostatics**	**Probability**	Used in this work
Scalar-valued	$\varphi(\mathbf{x})$	$\log p(\mathbf{x})$	-
Vector-valued	$\nabla\varphi(\mathbf{x})$ $\mathbf{E} = -\nabla\varphi(\mathbf{x})$ $\mathbf{F} = q'\mathbf{E} = -q'\nabla\varphi(\mathbf{x})$	$\nabla \log p(\mathbf{x})$	$\mathbf{F} \propto p(\mathbf{x})$ or $\log p(\mathbf{x})$
Scalar-valued	$\Delta\varphi(\mathbf{x}) = \nabla \cdot \nabla\varphi(\mathbf{x})$	$\mathrm{tr}(\nabla^2 \log p(\mathbf{x}))$	-

Table 3: Correspondence of concepts in electrostatics and probability.

For example, if we assume $\varphi(\mathbf{x}) = k\nabla \log p(\mathbf{x})$, then $\mathbf{F} = -q\mathbf{E} = -q\nabla\varphi(\mathbf{x}) = -qk\nabla \log p(\mathbf{x})$, which is similar to the driving force term $k(\cdot,\cdot)\nabla \log p(\mathbf{x})$ in SVGD. Therefore, in order to calculate

the force $\mathbf{F}$, we only need to gain knowledge about $\nabla\varphi(x)$, which says, equivalently in probability, that we only need to know $\nabla \log p(\mathbf{x})$, the gradient field can be conveniently estimated by techniques such as *score matching* Hyvärinen (2005).

K Discussions

EParVI is built on the principles that charges repel and attract each other, in regions of uniform density, charges are distributed uniformly; in regions where density varies, the clustering of negative charges follows the density distribution. The repulsive forces between negative charges ensure particle diversity and avoid mode collapse. These principles ensure the correctness and diversity of the inferred geometry. From an optimisation point of view, each update reduces global force imbalance by repositioning the particles in accordance with the force field Schmaltz et al. (2010); from an energy perspective, we are implicitly minimizing the overall electric potential which serves as the variational objective. Energy is minimized when forces are in equilibrium (i.e. system is electrically neutral)Teuber et al. (2011).

Computation in high-dimensions EParVI suffers from curse of dimensionality. The quality of sampling and inference results relies on the granularity of meshing. Finer mesh grid is needed for complex geometries (e.g. with large curvature). However, the computational complexity increases exponentially with dimension m. Efforts can be saved by, e.g. pre-calculating the probability matrix at grid points, as well as pairwise forces between cells, and storing them for later query. Fast Fourier transform techniques Gwosdek et al. (2014) have also been developed. It would be beneficial to leverage the fast inference capability of a trained neural network to approximate the force field as well.

Local optima This can lead to ambiguous distribution of the negative charges; adding perturbations to the particle in each iteration may help, e.g. projecting the particle onto the closest horizontal or vertical line connecting the grid points Schmaltz et al. (2010). We propose injecting some random noise to the move proposal, e.g. every k steps, to avoid getting stuck in local optima:

$$\mathbf{x}_j^{t+1} = \mathbf{x}_j^t + \tau' \Delta \mathbf{x}_j^t + \epsilon^t, \text{ every } k \text{ steps} \tag{64}$$

where $\epsilon_j^t \sim \mathcal{N}(0, \sigma^2)$. This is equivalent to saying $\mathbf{x}_j^{t+1} \sim \mathcal{N}(\mathbf{x}_j^t + \tau' \Delta \mathbf{x}_j^t, \epsilon_j^t)$. The proposed move $\tau' \Delta \mathbf{x}_j^t = \tau \mathbf{F}_j^t$ for the Euler scheme, and $\Delta \mathbf{x}_j^t = \mathbf{F}_j \Delta t^2 + \Delta \mathbf{x}_j^t$ for the Verlet and damped Verlet schemes.

Probabilistic move If, in an iteration, the particle takes a large step and jumps to a low density region, it may get trapped in that region due to small drag force (e.g. the extra regularisation forces outweighing the attracting force Schmaltz et al. (2010)). We thus suggest taking a probabilistic move to avoid occasionally large move, and assess the quality of each move via an MH step to decide if this move is to be executed:

$$\mathbf{x}_j^{t+1} = \begin{cases} \mathbf{x}_j^t + \tau \Delta \mathbf{x}_j^t, & \text{with probability } p_{accept} \\ \mathbf{x}_j^t, & \text{else} \end{cases} \tag{65}$$

where

$$p_{accept} = min\{1, \frac{p(\mathbf{x}_j^t + \tau \Delta \mathbf{x}_j^t)}{p(\mathbf{x}_j^t)}\} \tag{66}$$

in which the ratio doesn't require a fully know density. Using this acceptance-rejection criteria, we can then decide if a move proposal $\tau \Delta \mathbf{x}_j^t$ should be accepted or not. The use of an MH step, however, is two-side: while it prevents the particle of consideration from taking dangerous move in the current iteration, it may break the strategy for promoting global force equilibrium. It also induces extra cost.

EParVI diagnostics First, we need to ensure positivity of the charge magnitudes q_i, where $i \in \{1, 2, ...M^{neg}\}$, and $q_{i'}$, where $i' \in \{1, 2, ...M^{pos}\}$. Negative magnitudes can result, e.g. in scenarios where $q_{i'} = \log p(\mathbf{x}_{i'})$ (e.g. the LV dynamical system inference problem). Therefore, adjustment (offset) is needed. Further, there may be other constraints imposed by the problem itself, e.g. positive trajectories in the LV dynamical system inference problem, which requires special treatment (e.g.

$q_{i'}$=0) of the positive charge magnitudes at grid points yielding negative $(x(t), y(t))$ values. In each step, we monitor the number of free-moving negative charges which are sitting within the initialisation regime; large number of negative charges falling out of this region may indicate potentially wrong specification of the probabilistic model, or erroneous calculation of the positive charge magnitudes (e.g. negative values appear in magnitudes), or simply the repulsive forces is way bigger than the attracting force (i.e. too small positive charge magnitudes). Correctness of the probabilistic model can be checked via other sampling methods such as MCMC, or by comparing the MAP estimate with point ground truth values (if ground truth values are available). The imbalance between repulsive and attracting forces can be checked by running the simulation for one step, monitoring both values and increasing/decreasing the charge magnitudes accordingly.

Hyperparameters Hyper-parameters used in the SPH method include: number of positive charges (number of mesh grid points) M^{pos}, number of negative charges M^{neg}, negative charge magnitude (normally set to be 1.0), maximum positive charge magnitude q, total number of iterations T, position updating parameters τ, τ' and Δt, noise level σ, and others (e.g. annealing parameters).

EParVI doesn't have many parameters to tune - most of them can be empirically specified. Number of positive charges M^{pos} depends on the granularity to be achieved, it relies on dimension of the problem m, and it has great impact on the inference accuracy. It is set to be $M^{pos} = M^m$ if regular, equidistant mesh grid is used. M^{neg} can impact the inference quality as well, in general it can be much smaller than M^{pos}, and there is no hard requirement on it. the maximum positive charge magnitude q can be conveniently chosen by trial and error, one can choose it to make repulsive and attracting forces are on the same order in a trial step. T should be large enough to ensure a transient or steady state which is close to the target (one can monitor some metrics such as iteration-wise density or clustering metric to decide when to terminate). Small values for Δt, τ and τ' are required; adaptive schemes can be designated, i.e. in regions where the attracting force is large (i.e. high density regimes), smaller move is expected. One can also use different values for different dimensions (otherwise one dimension may converge fast while other dimensions not). For example, if the parameters to be inferred are of different scales, e.g. $a, d \in [0, 1]$ while $b, c \in [0, 0.05]$ in the LV system, we can use different Euler proportionality constant τ and different step sizes Δt and damping constant τ' for a, d and b, c. The noise level σ can perturb particle positions to avoid getting trapped in local optimal configuration (Eq.64); it is normally set to be small as compared to the move. There are other implementation tricks, as listed in Appendix.H, to facilitate the inference procedure.

Effect of mesh grid It is reported by Schmaltz et al. (2010) that, hexagonal structure is the optimal structure in 2D for achieving equilibrium; however, this cannot be well represented on the rectangular grid. If the rectangular grid is to be used, unpleasant artefacts such as multiple particles cluster at the same grid position can be resulted. This can be solved by constraining the particles to concentrate on moving towards nearest grid point, which can be realised by adding an extra, distance-inverse force component to the force acting on each particle (Eq.1s) which regularizes large movement from nearby grid points.

A temporal graph network perspective Dynamic movement of a group of interacting particles can be viewed as the evolution of a connected graph network over time. One can use network analysis techniques to identify the important contributing edges for each node at at any cross-section of time.

Insights from molecular dynamics Highly accurate and efficient first-principle based molecular simulations provide insights as well as a wealth of training data for modelling IPS Duignan (0). For example, accurate quantum chemistry algorithms, e.g. density functional theory (DFT), have made it possible to build fast and highly accurate models of water Bore & Paesani (2023). Neural network potentials (NNPsDuignan (0)), for example, learn the interaction between molecules based on quantum chemistry data and is capable of predicting the atomic potential field in a molecular system. It can capture the complex interaction patterns between molecules with high computational efficiency ($\mathcal{O}(M)$) and accuracy. These insights can be brought to create analogy when devising a particle system.

Sample clustering for multi-modal inference In the post-sampling stage, one may be interested in empirically estimating the statistics of $p(\mathbf{x})$ for use of e.g. *posterior predictive* in Bayesian modelling. Summarising the statistics of a distribution is not easy, particularly when it's high-dimensional and

multi-modal. With the (quasi) samples drawn, we can perform e.g. clustering to group the samples and summarise the statistics in each cluster. This can be achieved using available clustering and partition algorithms such as *k-d-tree*.

Learning $p(x)$ or $\log p(x)$ In some cases, directly learning $p(x)$ can be numerically unstable, especially when $p(x)$ is very small or large (e.g. chain products of likelihoods). Working with $\log p(x)$ helps to mitigate these issues because the log-transformation compresses the range of values, leading to more stable numerical computation. Also, the gradients of the log-probability $\nabla \log p(x)$ are more stable and easier to handle numerically than the gradients of p(x) itself, and therefore used commonly in ParVI (e.g. SVGD). However, for ParVI methods, if one decide to perform inference on $\log p(x)$, then a density estimator is needed to recover $p(x) = e^{\log p(x)}$. This is the case with Section.I.

Change of variables For vector-valued random variable x' and x who live in the same dimension $\mathbb{R}^m$, the *change of variables* principle applies:

$$p(x) = p'(x')/|detJ(x')| = p'(x')/|det(\frac{\partial x}{\partial x'})| = p'(x')/|det[\frac{\partial f_\theta(x')}{\partial x'}]| \text{ with } x = f(x') \tag{67}$$

where $|det[\frac{\partial f_\theta(x')}{\partial x'}]|$ is the absolute value of the determinant of the Jacobian matrix $Jf_\theta(x_i), i = 1, 2, ...M$. If a density is transformed multiple times, the chain rule of product can be used to arrive at the final transformed pdf, yielding a *normalizing flow* [22]. The EParVI inference algorithm for change of variables is presented in Algo.2. Note that, It requires computing the determinant of the Jacobian matrix of the transform, which can be computationally expensive for high dimensions.

Algorithm 2 EParVI inference for change of variables

- **Inputs**: target probability density function $p(x)$ with $x \in \mathbb{R}^m$; probability density function $p'(x')$ with $x' \in \mathbb{R}^m$. Number of particles M.
- **Outputs**: Particles $\{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^M$ whose empirical distribution approximates the target density $p(\mathbf{x})$.

1. *Generate samples in low dimensions*. Use EParVI to generate sample particles $x'_i \sim p'(x')$, $i = 1, 2, ...M$.

2. *Neural normalizing flow training*. Initialize a neural network f_θ: $\mathbb{R}^m \to \mathbb{R}^m$ which accepts x' and outputs x. (1) For each particle i, compute $x_i = f_\theta(x'_i)$ and individual loss $\mathcal{L}_i = -\log p(x_i) - \log|\det Jf_\theta(x'_i)|$.
(2) Compute mean loss $\frac{1}{M}\sum_{i=1}^M \mathcal{L}_i$. Back-propagating the error to update network weights.

2. *Inference and original sample reconstruction*. Mapping x'_i to x_i. (1) For each particle $x'_i, i = 1, 2, ..., M$, project it using the trained neural network: $x_i = f_\theta(x'_i)$.
(2) Perform density estimation (e.g. KDE) to obtain $\hat{p}'(x'_i)$.
(3) Estimate target density $\hat{p}(x_i) = \hat{p}'(x'_i)/|detJf_\theta(x'_i)|$.

3. *Return* $x_i, i = 1, 2, ..., M$ and the estimated density $\hat{p}(x)$.

Generalise to high dimensions Eq.67 requires a bijective and differentiable map $x = f_\theta(x')$. If we have the low dimensional samples $x'_i \in \mathbb{R}^{m'}$, e.g. produced by EParVI, and we want to project them into higher dimensional samples $x_i \in \mathbb{R}^m$, with $m' < m, i = 1, 2, ...M$. That is, we first generate samples $x' \sim p'(x')$ in low dimensions, which is feasible using EParVI, and then project the low-dimensional samples into high dimensions via $x = f_\theta(x')$, obtaining $x \sim p(x)$ and preserving the probability structure. The change of variables principle can no longer be applied in

[22] Normalizing flows Rezende & Mohamed (2016a) allow for learning a complex transformation between probability distributions.

this case (the Jacobian matrix is non-square). Instead, we employ a neural network to represent this non-linear mapping $x = f_\theta(x')$, with the training objective [23] $\mathcal{L}(x'_i) = -\log p(x_i) - \mathcal{L}'_i$, where $\mathcal{L}'_i$ is a measure of the contribution of the projected sample x_i to the diversity of $p(x)$ in the projected space $\mathbb{R}^m$, ensuring the mapping properly captures the structure of the target pdf $p(x)$. The algorithm is presented in Algo.3.

Algorithm 3 Projected EParVI inference

- **Inputs**: target probability density function $p(x)$ with $x \in \mathbb{R}^m$; probability density function $p'(x')$ with $x' \in \mathbb{R}^{m'}$. Number of particles M.
- **Outputs**: Particles $\{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^M$ whose empirical distribution approximates the target density $p(\mathbf{x})$.

1. *Generate samples in low dimensions*. Use EParVI to generate sample particles $x'_i \sim p'(x')$, $i = 1, 2, ...M$.

2. *Neural network training*. Initialize a neural network f_θ: $\mathbb{R}^{m'} \to \mathbb{R}^m$ which accepts x' and outputs x. (1) For each particle i, compute x_i = $f_\theta(x'_i)$ and individual loss $\mathcal{L}i = -\log p(x_i) - \mathcal{L}'_i$.
(2) Compute mean loss $\frac{1}{M}\sum_{i=1}^M \mathcal{L}_i$. Back-propagating the error to update network weights.

2. *Inference and original sample reconstruction*. Mapping $x'_i \in \mathbb{R}^{m'}$ to $x_i \in \mathbb{R}^m$. (1) For each particle $x'_i, i = 1, 2, ..., M$, project it using the trained neural network: $x_i = f_\theta(x'_i)$.
(2) Perform density estimation (e.g. KDE) to obtain $\hat{p}(x)$.

3. *Return* $x_i, i = 1, 2, ..., M$ and the estimated density $\hat{p}(x)$.

Algo.3 provides a way to estimate high-dimensional distributions by working in a lower-dimensional space, which can be more computationally efficient. Quality of the final estimation depends on the expressiveness of the neural network, design of the loss function and the training process. The diversity loss $\mathcal{L}'_i$ can be designed using strategies for example: (1) Product of singular values of the Jacobian. This gives a measure of the 'volume change': $\mathcal{L}'_i = \sum_{j=1}^{m'} \log \sigma_j[Jf_\theta(x'_i)]$, where σ_j are the singular values of the Jacobian Jf_θ. This approach accounts for expansion in all directions, and it is computationally expensive (routinely, SVD is performed to obtain the sigular values). One can use only the largest singular value which gives an upper bound on the expansion. (2) Project the Jacobian onto an orthonormal basis in the higher-dimensional space: $\mathcal{L}'_i = \log|\det[Jf_\theta(x'i)^T Jf_\theta(x'_i)]|^{1/2}$. (3) Use the trace of the Gram matrix of the Jacobian as a measure of magnitude: $\mathcal{L}'_i = \log \mathrm{Tr}[Jf_\theta(x'i)^T Jf_\theta(x'_i)]$. (4) Use the Frobenius norm of the Jacobian as a simple scalar measure: $\mathcal{L}'_i = \log|Jf_\theta(x'_i)|_F$. Each of these loss designs provides a way to capture the transformation's properties, by quantifying the 'expansion' or 'contraction' of the transformation without requiring a square Jacobian.

[23] Note that, regularization term can be added to promote sample diversity and smooth mapping.