DIV, GRAD, CURL, AND ALL THAT: ELECTROSTATIC MODELS FOR SCORE MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

We show how to estimate a generative model from samples by parameterizing its score function as the electric field of a static charge distribution. The score function is given by the gradient of the model's log-density, and as such, it must define a conservative vector field. By modeling the score as an electric field, we not only guarantee that it satisfies this curl-free constraint; we can also efficiently compute its divergence, via Maxwell's equations, from a local charge density. We parameterize the score as a superposition of electric fields from spherically symmetric charge distributions and uniformly charged slabs. We estimate the coefficients in these superpositions by minimizing a Fisher divergence, and notably, the optimal coefficients are given by the closed-form solution of a regularized least-squares problem. On one hand, these models have a natural connection to energy-based models and product-of-expert models, and they include existing kernel estimators as a special case; on the other hand, they can be learned without recourse to iterative gradient-based methods. We present the results of experiments on synthetic data sets as well as problems of higher dimensionality. Our results show that these charge-based models perform competitively with neural and kernel-based baselines, and they do so while using fewer parameters, learning more efficiently, and yielding more geometrically interpretable densities.

1 INTRODUCTION

Score matching is a powerful framework for learning probabilistic models without the need for evaluating their normalizing constants (Hyvärinen, 2005; 2007). Instead of directly modeling a probability distribution, score matching focuses on estimating its score function—the gradient of the log-density with respect to the data $\boldsymbol{s}(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$. Because normalizing constants disappear under differentiation, this allows generative models to be learned even when maximum likelihood is impractical. The score also identifies the direction of increase of the log-density, making it useful for sampling and denoising. Score matching consequently plays an important role in energy-based modeling, score-based generative modeling, and diffusion models (LeCun et al., 2006; Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021a;b).

To learn the score from samples, score matching minimizes the Fisher divergence (Hyvärinen, 2005) between the true score and a parameterized model. The resulting objective function depends on both the magnitude of the modeled score $\boldsymbol{s}_\theta(\boldsymbol{x})$ and its divergence $\nabla \cdot \boldsymbol{s}_\theta(\boldsymbol{x})$. While it seems natural to parameterize the score with a neural network, a fundamental tension arises in doing so. On one hand, we could model the negative log density using a neural network and differentiate it to obtain a *conservative* score field. However, evaluating its divergence requires an expensive Hessian backpropagation or a noisy stochastic trace estimation (Song et al., 2020). On the other hand, we could use a neural network to directly model the score. While this gives a flexible vector field, it does not guarantee that it is conservative (Salimans & Ho, 2021; Chao et al., 2022). These considerations motivate the following question:

> How can we construct flexible, conservative score models while providing a simple, analytical divergence?

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Electrostatics provides a natural connection between these two requirements (Schey, 1996). Here, a charge density $q(\boldsymbol{x})$ describes how positive and negative charge is distributed in space. It generates an electric field $\boldsymbol{E}(\boldsymbol{x})$, which gives the direction and magnitude of the force on a positive charge at $\boldsymbol{x}$. Maxwell's equations (Griffiths, 2005) guarantee that an electrostatic field is conservative and relate its divergence to the local charge density, $\nabla \cdot \boldsymbol{E}(\boldsymbol{x}) = q(\boldsymbol{x})$. Choosing a charge distribution with analytic electric fields leads to both a conservative score model and an explicit expression for the divergence that is required by score matching.

In this work, we construct a family of score models from electrostatic charge configurations. Specifically, we parameterize the score field as a superposition of electric fields generated by simple components. Importantly, adding charge distributions adds their corresponding fields and potentials, so the combined score remains conservative, and its divergence remains directly computable, given by the combined charge density. Specifically, we parameterize the score field as a superposition of electric fields generated by charged slabs and balls. These geometries have the capability to model arbitrary score functions and provide complementary structured fields: balls model local radial variation, and slabs model variation along selected directions.

To train the electrostatic models, we minimize the Fisher divergence between the unknown score function and the modeled electric field. We show that its empirical objective becomes a balance between the electric field energy and the magnitude of the underlying charge densities. With the locations, subspaces, and scales of the charge distributions fixed, the score is linear in their coefficients. Optimization over these coefficients therefore reduces to the closed-form solution of a regularized linear system, avoiding iterative gradient descent over the score model. In summary, superposition gives us both a flexible way to build models and also a simple procedure to estimate its components.

Our electrostatic formulation includes, as a special case, finite-basis kernel exponential families when kernel functions are used as scalar potentials (Sriperumbudur et al., 2017; Sutherland et al., 2018; Zhou et al., 2020; Wenliang et al., 2019). In particular, Gaussian kernel potentials generate the spherically symmetric fields in the electrostatic configuration. Slab components extend this dictionary with fields that vary along selected directions.

We evaluate how the choice of charge geometry affects score estimation on known-score synthetic distributions and higher-dimensional problems. Our experiments compare spherically symmetric charges, uniformly charged slabs and mixed-geometry models; study performance as the dimension and dictionary size changes; and compare performance with neural and kernel-based baselines on known-score synthetic distributions and in full MNIST and CIFAR-10 WAE training.

2 Electrostatic Formulation of Score Matching

Let $p(\boldsymbol{x})$ be a probability density on $\boldsymbol{x} \in \mathbb{R}^d$ with score $\boldsymbol{s}(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$. In many applications, rather than estimating $p(\boldsymbol{x})$ directly from samples, it is both sufficient and more efficient to estimate the score $\boldsymbol{s}(\boldsymbol{x})$ instead; this is the essential idea of *score-matching*. But a fundamental tension arises in this approach when the score is parameterized by neural networks or other similarly complicated models. We show how this tension arises in Section 2.1, and we show how it is elegantly resolved by electrostatic models in Section 2.2.

2.1 Score-matching with the Fisher divergence

Suppose the score is parameterized as $\boldsymbol{s}_\theta(\boldsymbol{x})$ for some expressive family of models with parameters θ (e.g., the weights in a neural network). In principle, we could estimate θ by minimizing the *Fisher divergence* (Hyvärinen, 2005):

$$\mathcal{J}(\boldsymbol{s}_\theta) = \mathbb{E}_p \left[\|\boldsymbol{s}_\theta(\boldsymbol{x}) - \nabla \log p(\boldsymbol{x})\|^2 \right]. \tag{1}$$

Although the true score is unknown, this objective can be rewritten in a form that only depends on samples from p. Expanding the square in eq. 1 and integrating by parts, we obtain

$$\mathcal{J}(\boldsymbol{s}_\theta) = \mathbb{E}_p \left[\|\boldsymbol{s}_\theta(\boldsymbol{x})\|^2 \, + \, 2\nabla \cdot \boldsymbol{s}_\theta(\boldsymbol{x}) \right] \, + \text{const.}, \tag{2}$$

where we assume standard smoothness and vanishing-boundary conditions in eq. 2. The first two terms can be easily estimated if we have samples $\{\boldsymbol{x}_i\}$ from the density $p(\boldsymbol{x})$. In particular, replacing

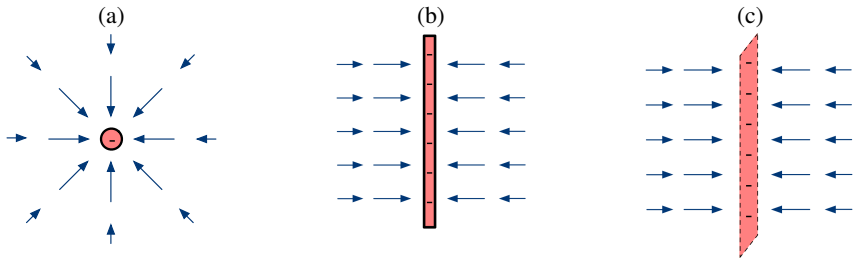


Figure 1: Electric field vectors arising from negative charges: (a) a point at the origin, $\boldsymbol{E}(\boldsymbol{x}) \propto -\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|^d}$; (b) a uniformly charged infinite line, $\boldsymbol{E}(\boldsymbol{r}, z) \propto -\frac{\boldsymbol{r}}{\|\boldsymbol{r}\|^{d-1}}$; (c) a uniformly charged infinite plane, $\boldsymbol{E}(\boldsymbol{x}) \propto -\operatorname{sgn}(\boldsymbol{u}^\top \boldsymbol{x} - b)\boldsymbol{u}$, where $\boldsymbol{u}$ is unit a normal and $\operatorname{sgn}(\cdot)$ returns ± 1 depending on the side of the plane. The line and plane extend infinitely in parallel directions beyond the displayed regions. The formulas apply away from the charges.

the expectations over p in eq. 2 by empirical averages, we obtain the score-matching objective

$$\widehat{\mathcal{J}}(\boldsymbol{s}_\theta) = \frac{1}{n}\sum_{i=1}^{n}\Big[\|\boldsymbol{s}_\theta(\boldsymbol{x}_i)\|^2 + 2\nabla\cdot\boldsymbol{s}_\theta(\boldsymbol{x}_i)\Big]. \tag{3}$$

Notice that this objective function requires both the score and its divergence. While it seems quite natural to minimize eq. 3 over a highly expressive family, an essential tension that arises if, for instance, $\boldsymbol{s}_\theta(\boldsymbol{x})$ is parameterized by a neural network. There are two obvious approaches to proceed in this fashion. On one hand, we might use a neural network to parameterize the negative log-density $V_\theta(\boldsymbol{x}) = -\log p(\boldsymbol{x})$ and express the score as its negative gradient $\boldsymbol{s}_\theta(\boldsymbol{x}) = -\nabla V(\boldsymbol{x})$. This guarantees a conservative field but its divergence is the Laplacian $-\Delta V_\theta(x)$, which can be expensive to evaluate in general. Alternatively, we might parameterize $\boldsymbol{s}_\theta(\boldsymbol{x})$ directly, but this does not automatically ensure that it is the gradient of a scalar function. So, our goal is to construct models that combine conservativeness with an efficiently computable divergence.

2.2 Score-matching with electrostatics

Electrostatics provides a natural connection between these two requirements, giving a way to construct conservative score models whose divergences are easy to compute. We begin by introducing the physical quantities underlying this connection.

A *charge density* $q(x)$ describes how electric charge is distributed in space. It is a scalar field over $\mathbb{R}^d$ whose integral yields the total amount of charge in all of space. Note that charges can take either positive or negative sign. This density generates an *electric field* $\boldsymbol{E}(\boldsymbol{x})$, a vector that indicates the direction and magnitude of the force experienced by a positive charge at $\boldsymbol{x}$ due to the charge density q. Figure 1 shows the electric fields in $\mathbb{R}^3$ that arise from certain canonical charge densities, such as a negative point charge (*left*), a uniform infinite line charge (*middle*), and a uniformly charged infinite plane (*right*). These fields point inward because opposite charges attract. In electrostatics, the charge density q is assumed to be stationary, or in other words, there are no electrical currents.

Electrostatic fields are governed by Maxwell's equations. In $\mathbb{R}^3$, these equations take the form

$$\nabla\times\boldsymbol{E}(\boldsymbol{x}) = \boldsymbol{0}, \qquad \nabla\cdot\boldsymbol{E}(\boldsymbol{x}) = q(\boldsymbol{x}), \tag{4}$$

where the charge density $q(\boldsymbol{x})$ is measured in units such that the vacuum permittivity constant is unity. The first equation states that the electric field is *curl-free*. The second equation, known as *Gauss's law*, states that the divergence of the field at each point equals the charge density there. In d dimensions, the curl-free condition generalizes to the constraint that $\frac{\partial E_i}{\partial x_j} = \frac{\partial E_j}{\partial x_i}$ for all $1 \le i < j \le d$. From these equations we make two impactful observations for score-matching.

First, static electric fields are inherently conservative. They can be expressed as the negative gradient of a scalar potential $V(\boldsymbol{x})$, or the *electric potential*: that is,

$$\boldsymbol{E}(\boldsymbol{x}) = -\nabla V(\boldsymbol{x}). \tag{5}$$

Table 1: Correspondence between electrostatics and score matching in machine learning.

Concept	Electrostatics	Machine Learning
Potential	$V(\boldsymbol{x})$	$p(\boldsymbol{x}) \propto e^{-V(\boldsymbol{x})}$
Field	$\boldsymbol{E}(\boldsymbol{x}) = -\nabla V(\boldsymbol{x})$	$\nabla \log p(\boldsymbol{x})$
Charge	$q(\boldsymbol{x}) = \nabla \cdot \boldsymbol{E}(\boldsymbol{x})$	$\Delta \log p(\boldsymbol{x})$
Superposition	$\boldsymbol{E}(\boldsymbol{x}) = \sum_k \theta_k \boldsymbol{E}_k(\boldsymbol{x})$	$p(\boldsymbol{x}) \propto \prod_k p_k(\boldsymbol{x})^{\theta_k}$

Physically, the electric potential describes the amount of work required to transport a unit positive charge to $\boldsymbol{x}$ from (say) an infinite distance away. *Thus, it is natural to identify the score of a probability density with a static electric field, and its negative log-density of $\boldsymbol{x}$ with the electric potential* $V(\boldsymbol{x})$ (up to some additive constant).

Second, the divergence of the electric field at $\boldsymbol{x}$ is given simply by the charge density at $\boldsymbol{x}$. Gauss's law suggests an elegant solution to the dilemma posed earlier—whether, on one hand, to parameterize the score directly as the gradient of a potential (whose divergence may be difficult to compute), or on the other hand, to parameterize the score directly as a vector field that is not necessarily conservative. *The answer is not to parameterize the score directly, but to parameterize the charge density instead.* We can then draw upon an extensive body of physical understanding for simple charge distributions, and their *superpositions*, to construct conservative fields with tractable divergences.

To formalize this approach, suppose that we parameterize a charge density $q_\theta(\boldsymbol{x})$ over $\mathbb{R}^d$ and identify the score with the electric field $\boldsymbol{s}_\theta(x) = \boldsymbol{E}_\theta(x)$. Then from eq. 4, we see that the score satisfies $\nabla \cdot \boldsymbol{s}_\theta(\boldsymbol{x}) = q(\boldsymbol{x})$, and the corresponding potential satisfies the Poisson equation $-\Delta V_\theta(\boldsymbol{x}) = q_\theta(\boldsymbol{x})$. Moreover, since the Poisson equation can be solved via Green's functions under mild regularity and boundary conditions (Griffiths, 2005), we see that any score can be represented as the electric field of a static charge density. Finally we write the empirical Fisher divergence in eq. 3 fully in terms of electric fields and the charge densities that generate them. Identifying $\boldsymbol{s}_\theta(\boldsymbol{x}) = \boldsymbol{E}_\theta(\boldsymbol{x})$, we obtain the score-matching objective

$$\widehat{\mathcal{J}}(\boldsymbol{E}_\theta) = \sum_{i=1}^{n} \left[\left\| \boldsymbol{E}_\theta(\boldsymbol{x}_i) \right\|^2 + 2 q_\theta(\boldsymbol{x_i}) \right]. \tag{6}$$

This score-matching objective will define a conservative electric field (i.e., score) whenever $\boldsymbol{E}_\theta(\boldsymbol{x})$ is determined by a charge density via Maxwell's equations in eq. (4).

Now consider how to parameterize the charge density to reap these benefits. Suppose we assemble a dictionary of charge densities $\{q_k(\boldsymbol{x})\}_{k=1}^K$ over $\mathbb{R}^d$ for which we have solved Poisson's equation to determine their corresponding electric potentials $\{V_k(\boldsymbol{x})\}_{k=1}^K$ and electric fields $\{\boldsymbol{E}_k(\boldsymbol{x})\}_{k=1}^K$. By superposition of these charge densities, we obtain

$$q_\theta(\boldsymbol{x}) = \sum_k \theta_k q_k(\boldsymbol{x}). \tag{7}$$

Then, given samples $\{\boldsymbol{x}_i\}$ in $\mathbb{R}^d$, we fit the coefficients by minimizing the *least-squares problem*

$$\min_\theta \left\{ \sum_{i=1}^{n} \left[\left\| \sum_k \theta_k \boldsymbol{E}_k(\boldsymbol{x}_i) \right\|^2 + 2 \sum_k \theta_k q_k(\boldsymbol{x_i}) \right] \right\}. \tag{8}$$

This is a much simpler optimization than, say, what is required if the score or the log-density is parameterized by a neural network. Our goal is to explore charge densities (and hence scores) that can be parameterized and estimated in this way.

Table 1 summarizes the correspondence between electrostatic quantities and their machine learning counterparts. We note that superposition (eq. 7) also gives the associated density of the form

$$p(\boldsymbol{x}) \propto \exp\left(-\sum_k \theta_k V_k(\boldsymbol{x}) \right) = \prod_k \exp(-\theta_k V_k(\boldsymbol{x})), \tag{9}$$

which is a so-called *product-of-experts* model, where the kth expert is geometrically weighted by θ_k. Eq. 9 suggests another constraint that the charge densities $q_k(\boldsymbol{x})$ must satisfy if they are to be used for score-matching; they must be designed so that the resulting density in eq. 9 is integrable. In the next section, we assemble a dictionary of charge densities with this property.

3 Charge Distributions, Score Fields, and Estimation

Kernel exponential families (KEFs) are a common approach to score matching problems (Sriperumbudur et al., 2017; Sutherland et al., 2018; Wenliang et al., 2019). For example, a finite Gaussian KEF writes its unnormalized log-density as

$$\log \widetilde{p}(\boldsymbol{x}) = \log p_0(\boldsymbol{x}) + \sum_{k=1}^{K} a_k \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_k\|^2}{2\sigma_k^2}\right). \tag{10}$$

Its score is the gradient of this expression. In the electrostatic interpretation of this fundamental Gaussian KEF model, each Gaussian kernel contributes an electric field obtained from a Gaussian potential: the negative log-density plays the role of electric potential, the score is its electric field, and the divergence of the score is the associated charge density. Thus, a finite Gaussian KEF can be viewed as a superposition of spherically symmetric charge fields.

This viewpoint suggests considering constructing score estimation models from their electric charges, and thereby considering charge distributions including and beyond the spherically symmetric charge derived from the Gaussian KEF model. In this model, we consider the Gaussian KEF spherically symmetric components, which produce local radial fields, and uniformly charged slabs, which produce fields that vary along a chosen direction. These are visualized in Figure 2, and are generalizations of the point, line, and plane configurations illustrated in Figure 1.

Both satisfy $\boldsymbol{E}_k = -\nabla V_k$ and $q_k = \nabla \cdot \boldsymbol{E}_k = -\Delta V_k$. Consequently, any superposition of these fields is conservative, and its divergence is the corresponding superposition of charge densities. We formalize their charges, resulting fields, and relationships below.



Figure 2: Projected charge geometries in three dimensions. Left: the field varies radially in a spherically symmetric charge density; the charge extends beyond the displayed spherical region. Right: uniformly charged slab density. Red and blue indicate positive and negative charge density, respectively, and arrows indicate the resulting electric field. The fields extend beyond the displayed charge regions.

3.1 Spherically Symmetric Charge Densities

Consider a radially symmetric field around a center $\boldsymbol{c} \in \mathbb{R}^d$, which therefore decays with distance from $\boldsymbol{c}$ rather than ending at a sharp boundary. We define a component through its Gaussian potential:

$$\begin{aligned} V_{\text{sph}}(\boldsymbol{x}) &= \sigma^2 \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{2\sigma^2}\right), \\ q_{\text{sph}}(\boldsymbol{x}) &= \left(d - \frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{\sigma^2}\right) \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{2\sigma^2}\right), \\ \boldsymbol{E}_{\text{sph}}(\boldsymbol{x}) &= (\boldsymbol{x}-\boldsymbol{c}) \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{2\sigma^2}\right), \qquad \sigma > 0. \end{aligned} \tag{11}$$

The charge density changes q_{sph} sign at $r = \sigma\sqrt{d}$; it does not terminate at a spherical boundary and has zero net charge. Changing the sign of the fitted coefficient on a given spherically symmetric charge component reverses its field and charge.

The relationship to Gaussian KEFs is clear: the gradient of a Gaussian kernel equals $-\boldsymbol{E}_{\text{sph}}/\sigma^2$. With matched centers and widths of each component, a finite Gaussian KEF therefore belongs to the spherical-only version of our score family. Further discussion of different KEF models, which can be viewed as special cases of our electrostatic model, are in Section D.2.

3.2 Uniformly charged slabs

A uniformly charged slab has finite thickness and extends infinitely along the parallel hyperplanes that bound it. Let $\boldsymbol{u} \in \mathbb{R}^d$ be a unit normal. For a slab centered at $\boldsymbol{u}^\top \boldsymbol{x} - b = 0$, with half-width $L > 0$ and uniform charge density q_1, its potential, field, and charge density are

$$V_{\text{slab}}(\boldsymbol{x}) = \begin{cases} -\frac{q_1}{2}(\boldsymbol{u}^\top \boldsymbol{x} - b)^2, & \text{if} \quad |\boldsymbol{u}^\top \boldsymbol{x} - b| \le L, \\ -q_1 L|\boldsymbol{u}^\top \boldsymbol{x} - b| + q_1 L^2/2, & \text{if} \quad |\boldsymbol{u}^\top \boldsymbol{x} - b| > L, \end{cases} \tag{12}$$

$$\boldsymbol{E}_{\text{slab}}(\boldsymbol{x}) = \begin{cases} q_1(\boldsymbol{u}^\top \boldsymbol{x} - b)\,\boldsymbol{u} & \text{if} \quad |\boldsymbol{u}^\top \boldsymbol{x} - b| \le L \\ q_1 L\,\boldsymbol{u} & \text{if} \quad \boldsymbol{u}^\top \boldsymbol{x} - b > L \\ -q_1 L\,\boldsymbol{u} & \text{if} \quad \boldsymbol{u}^\top \boldsymbol{x} - b < -L, \end{cases} \tag{13}$$

$$q_{\text{slab}}(\boldsymbol{x}) = \begin{cases} q_1 & \text{if} \quad |\boldsymbol{u}^\top \boldsymbol{x} - b| \le L \\ 0 & \text{otherwise}, \end{cases} \tag{14}$$

Thus, the field varies linearly inside the slab and is constant outside it. The charge density and electric field divergence are exactly zero outside the slab. When fitting, we use one-sided boundaries as dictionary components:

$$V_{\boldsymbol{u},b}(\boldsymbol{x}) = -\tfrac{1}{2}[\boldsymbol{u}^\top \boldsymbol{x} - b]_+^2, \qquad \boldsymbol{\phi}_{\boldsymbol{u},b}(\boldsymbol{x}) = [\boldsymbol{u}^\top \boldsymbol{x} - b]_+ \boldsymbol{u}, \qquad \rho_{\boldsymbol{u},b}(\boldsymbol{x}) = \mathbf{1}\{\boldsymbol{u}^\top \boldsymbol{x} - b > 0\}. \tag{15}$$

Pairing two of these one-sided boundaries recovers a finite uniformly charged slab, as shown in Section A. This one-sided slab model is equivalent to a tied-direction ReLU MLP score parameterization and discussed in Section D.3.

3.3 Superposition of charge distributions

Let $(V_k, \boldsymbol{\phi}_k, \rho_k)$ denote the potential, field, and charge density of a spherical or slab-boundary component. We can combine a mixture of these components with a base linear field:

$$\begin{aligned} V_\theta(\boldsymbol{x}) &= -\tfrac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_0)^\top \boldsymbol{\Omega}(\boldsymbol{x} - \boldsymbol{x}_0) + \sum\nolimits_k \theta_k V_k(\boldsymbol{x}), \\ \boldsymbol{s}_\theta(\boldsymbol{x}) &= \boldsymbol{\Omega}(\boldsymbol{x} - \boldsymbol{x}_0) + \sum\nolimits_k \theta_k \boldsymbol{\phi}_k(\boldsymbol{x}), \\ q_\theta(\boldsymbol{x}) &= \operatorname{tr}(\boldsymbol{\Omega}) + \sum\nolimits_k \theta_k \rho_k(\boldsymbol{x}). \end{aligned} \tag{16}$$

Here, the base linear field $\boldsymbol{\Omega}$ is symmetric and corresponds to a spatially uniform charge distribution. Maxwell's equations are satisfied, and therefore relate the score model to its charge densities, for any fitted coefficients $\{\boldsymbol{\theta}\}$: $\boldsymbol{s}_\theta = -\nabla V_\theta$ and $q_\theta = \nabla \cdot \boldsymbol{s}_\theta$. The linear base field can represent the score of an arbitrary Gaussian distribution, and the additional charge components then add a nonlinear variation. When e^{-V_θ} is integrable, it also defines a probability density after normalization. Derivations of the component fields and potentials are given in Section A.

3.4 Quadratic coefficient estimation

A key advantage of the model in eq. 16 is that the score depends linearly on the coefficients θ_k, which can easily be optimized under the Fisher divergence. For fixed components of the score fields, substituting eq. 16 into the empirical objective in eq. 3 and collecting terms, we obtain the quadratic form

$$\widehat{\mathcal{J}}(\boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{G} \boldsymbol{\theta} + 2\boldsymbol{g}^\top \boldsymbol{\theta} + \text{const} \tag{17}$$

where $G_{kl} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top \boldsymbol{\phi}_l(\boldsymbol{x}_i)$, $g_k = \frac{1}{n}\sum_{i=1}^n \left(\boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top \boldsymbol{\Omega}(\boldsymbol{x}_i - \boldsymbol{x}_0) + \nabla \cdot \boldsymbol{\phi}_k(\boldsymbol{x}_i)\right)$. Here, $\rho_k = \nabla \cdot \boldsymbol{\phi}_k$ is the charge density of component k. When $\boldsymbol{G}$ is nonsingular, minimizing eq. 17 reduces to $\boldsymbol{G}\boldsymbol{\theta} = -\boldsymbol{g}$, giving the solution $\boldsymbol{\theta}^\star = -\mathbf{G}^{-1}\boldsymbol{g}$. These expressions apply to spherical and slab components, as well as their superpositions. The full derivation is given in Section B.1.

In practice, the Gram matrix $\mathbf{G}$ may be poorly conditioned. To stabilize the solve, we add ℓ_2 regularization to the objective, $\widehat{\mathcal{J}}_{\text{reg}}(\boldsymbol{\theta}) = \widehat{\mathcal{J}}(\boldsymbol{\theta}) + \alpha\|\boldsymbol{\theta}\|^2$, which yields the system $(\mathbf{G} + \alpha\mathbf{I})\boldsymbol{\theta} = -\boldsymbol{g}$ with solution $\boldsymbol{\theta}^\star = -(\mathbf{G} + \alpha\mathbf{I})^{-1}\boldsymbol{g}$. The symmetric linear base field is also fit. Its conditional updates and the joint refitting procedure are described in Sections B.2 and B.3.

We consider both full dictionaries and adaptive subsets. Adaptive fitting ranks and selects candidate charge component additions by their reduction of the regularized training objective, then refits the selected coefficients together with the linear base together. Validation then selects the dictionary size, widths, and regularization strengths. Sections C and E.3 give the construction and selection procedures used in each experiment.

4 Related Work

Score matching was introduced by Hyvärinen (2005; 2007) as a method for estimating unnormalized statistical models through the Fisher divergence, thereby avoiding intractable normalization constants. Since then, score matching has become a foundational tool in modern machine learning, including denoising score matching (Vincent, 2011; Alain & Bengio, 2014; Karras et al., 2022), score-based Langevin dynamics (Song & Ermon, 2019; 2020), diffusion models (Ho et al., 2020; Song et al., 2021a;b; Lu et al., 2022; Singhal et al., 2024; Song et al., 2020), energy-based models (Marlin et al., 2010; Swersky et al., 2011; Li et al., 2023; Salimans & Ho, 2021), and variational inference (Yang et al., 2019; Yu & Zhang, 2023; Cai et al., 2024b;a; 2025).

A line of related work studies structured parameterizations of vector fields. Neural score estimators, including sliced score matching (SSM) (Song et al., 2020), are flexible but do not automatically satisfy geometric constraints such as conservativeness or integrability (Chao et al., 2022); we further discuss the relationship to SSM (Song et al., 2020) in Section D.1. Several approaches therefore model scores through scalar energy functions, whose gradients define the score implicitly LeCun et al. (2006); Grathwohl et al. (2020). Our work follows this philosophy but imposes additional structure through electrostatics.

Kernel-based score estimators, including score matching with reproducing kernels Sriperumbudur et al. (2017); Sutherland et al. (2018); Wenliang et al. (2019), nonparametric score estimators Zhou et al. (2020), and spectral methods such as the spectral Stein gradient estimator (SSGE) Shi et al. (2018), provide strong nonparametric baselines in moderate dimensions. These methods often achieve high accuracy but may scale poorly with sample size or require expensive matrix decompositions and bandwidth tuning. Most closely related are estimators that, like ours, fix a basis and solve a regularized quadratic Fisher-divergence objective in closed form: the kernel exponential-family and nonparametric kernel estimators above, which work in an RKHS, and the least-squares log-density gradient (LSLDG) family Sasaki et al. (2014; 2018), which uses Gaussian-derivative features. By contrast, our proposed electrostatic model uses a mixture of charged slabs geometry that can be represented as a ReLU neural network with tied weights, giving an explicitly curl-free field with piecewise-constant divergence. We make the LSLDG comparison clear in Section D.4. More broadly, physics-inspired machine learning has shown that inductive biases derived from scientific principles can improve learning efficiency and interpretability. Examples include Hamiltonian neural networks (Greydanus et al., 2019), equivariant architectures (Cohen & Welling, 2016), deep generative models (Sohl-Dickstein et al., 2015), PDE-constrained learning methods (Raissi et al., 2019), and Poisson flow generative models (Xu et al., 2022; 2023). PFGM uses electrostatic fields induced by data charges in an augmented space to define a sampling flow, whereas we use charge distributions as a parameterization of the score itself and train them by Fisher divergence. Our contribution shows that electrostatics offers a physically-grounded language with charges as the fundamental source for score estimation: charge densities determine potentials, potentials determine fields, and fields coincide with scores.

5 Experimental Results

We evaluate the different charge configurations in electrostatic score models on synthetic distributions with known score and on latent score estimation in Wasserstein autoencoders (WAEs). We compare with SSGE (Shi et al., 2018), SSM (Song et al., 2020), and kernel-based estimators (Wenliang et al., 2019; Zhou et al., 2020; Sutherland et al., 2018). Model selection uses held-out score-

matching loss, without access to the true score. Data, fitting procedures, and baseline specifications are given in Section E.

5.1 Score estimation in 2D

We compare adaptive spherical, slab, and mixed dictionaries on Banana, GMM, and X-shape distributions. Each model uses the superposition in eq. 16; K counts the selected charge components. We use $1,600$ fitting, 400 validation, and $5,000$ test observations, and report mean test score MSE over five runs. Table 2 shows that the geometries have different strengths. The mixed model has the lowest GMM error (0.088 versus 0.121 of the best baseline), while slabs outperform the best baseline on X-shape (1.377 versus 1.541). The DKEF baseline remains strongest on Banana, with spherical components performing best of the electrostatic models. Figure 3 illustrates their selected charges and resulting score field on X-shape. Full-dictionary fits and complete baseline results are in Section F.

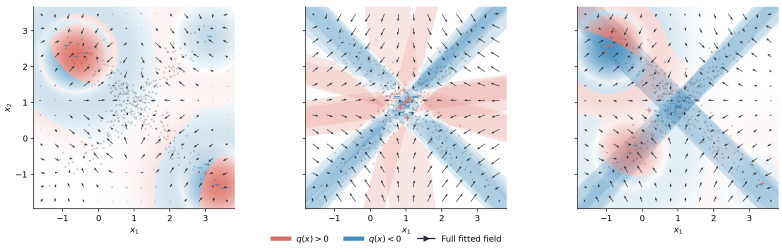


Figure 3: Selected charge components with spherical, slab, and mixed geometries on the X-shape distribution. Shading shows the fitted charge regions and arrows the resulting score field; only the largest-magnitude components are displayed. Additional visualizations are in Section F.4.

Dataset	Linear Base	Spherical	Slabs	Mixed Geometries	Best Baseline
Banana	1.317 ± 0.022 $K = 0$	0.102 ± 0.028 $K = 41.6$	0.292 ± 0.094 $K = 62.4$	0.234 ± 0.086 $K = 130.2$	$\mathbf{0.074 \pm 0.011}$ DKEF
GMM	1.845 ± 0.012 $K = 0$	0.142 ± 0.011 $K = 44.8$	0.132 ± 0.019 $K = 17.6$	$\mathbf{0.088 \pm 0.012}$ $K = 21$	0.121 ± 0.012 SSM
X-shape	14.890 ± 0.085 $K = 0$	2.049 ± 0.099 $K = 60.8$	$\mathbf{1.377 \pm 0.066}$ $K = 115.2$	1.583 ± 0.161 $K = 145.2$	1.541 ± 0.097 Curl-free IMQ

Table 2: Test score MSE in two dimensions (mean $\pm$ SE). K denotes the mean number of selected charge components. "Best Baseline" refers to the lowest MSE of the baseline models on the given synthetic distribution. Full baseline 2D results in Section F.1.

5.2 Higher-dimensional synthetic and real-data experiments

Synthetic dimension sweep. Motivated by the grid-distribution experiments of Sutherland et al. (2018); Zhou et al. (2020), we extend the four-component GMM by increasing the number of components with dimension. Our construction uses $2d$ equally weighted Gaussians such that the four original centers are retained from Section 5.1; details in Section E. We use the same $1,600$ fitting, 400 validation, and $5,000$ test observations at each dimension, with dimension $d \in [2, 100]$. We focus on the adaptive mixed model to allow both geometries within one estimator. Figure 4 reports five-seed test MSE; Banana and X-shape sweeps are in Section F.3.

The mixed model performs well relative to the baselines, particularly in the middle of the tested GMM dimension range. The additional Banana and X-shape sweeps in Section F.3 show comparable or lower error than the baselines across much of their dimension range.

Wasserstein autoencoders. Following Song et al. (2020), we estimate the aggregate posterior score from latent mini-batches during WAE training (Tolstikhin et al., 2018). This score is intractable, making WAE a good way to test score estimators downstream on MNIST and CIFAR-10. Table 3 reports test negative log-likelihood estimated by annealed importance sampling (Neal, 2001) on

MNIST latent dimensions $z \in \{8, 32, 64\}$ and CIFAR-10 latent dimensions $z \in \{8, 32\}$. See Section E for training and hyperparameter details.

Table 3 shows that the mixed electrostatic model has the lowest NLL on MNIST at $z = 8$ and improves on the linear model in all five settings. SSM has the lowest NLL at the remaining latent dimensions.

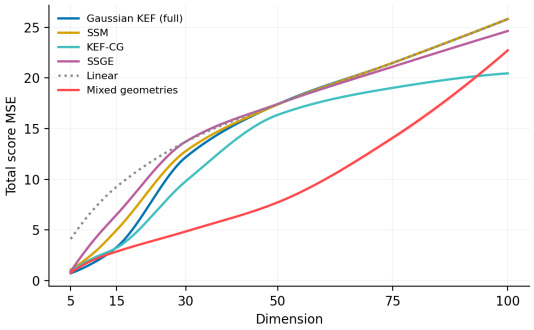


Figure 4: Test score MSE on the $2d$-component GMM. Curves show means over five seeds; shading denotes one standard error.

Method	MNIST $z=8$	MNIST $z=32$	MNIST $z=64$	CIFAR-10 $z=8$	CIFAR-10 $z=32$
Linear	98.54	94.62	97.18	79.68	76.27
Mixed	***98.10***	*92.17*	*95.89*	79.67	*74.08*
SSM	98.50	**91.20**	**93.93**	**79.27**	**72.00**
SSGE	98.79	92.21	96.06	80.01	74.64
KEF-CG	98.72	93.48	99.81	*79.62*	74.38

Table 3: WAE test NLL estimated by AIS (↓), with latent dimension z. Bolded is best result; italicized is best non-neural result.

At $d = 30$, the average CPU fit across five seeds took 0.213 s for adaptive mixed geometries, 0.717 s for KEF-CG, the fastest kernel method in this comparison, and 3.535 s for SSM. As expected, Linear is the fastest at 0.0002 s. Additional timing comparisons in Section G.5.



Figure 5: Generated MNIST samples from the WAE model at latent dimension $z = 8$ with our mixed geometries estimator. Additional dimension and baseline-generated visuals in Section F.4.

6 Discussion, Limitations, and Future Work

We presented an electrostatic perspective on score matching that models scores as electric fields generated by static charge distributions. This gives a physically motivated inductive bias: the learned score is curl-free by construction and has an analytic divergence. With spherically symmetric and uniform slab charge densities, the model remains interpretable and reduces Fisher-divergence training to a closed-form efficient solution, while remaining competitive with neural and kernel baselines.

A current limitation is that slab directions and spherical centers and widths are selected heuristically. Although coefficient fitting is quadratic for fixed geometries, current selection methods may not find globally optimal configurations and can be computationally expensive. Additionally, joint optimization of $\mathbf{\Omega}$ and $\boldsymbol{x}_0$ requires iterative Lyapunov solves. Our empirical evaluation includes known-score synthetic distributions through $d = 50$ and WAE experiments on MNIST and CIFAR-10.

Several other potential avenues remain for future work. First, it is straightforward to generalize the spherical and slab-like geometries to a generalized cylinder framework, where Gaussian spheres and Gaussian charged slabs are recovered by varying dimension of the subspace on which the charge density depends. This expands the geometries considered and may simplify the adaptive selection framework. Second, extending the framework to denoising score matching and diffusion-based generative models could provide a more scalable route for large-scale applications. Finally, further theoretical analysis of approximation bounds, generalization performance, and the role of regularization may help design new electrostatic score models.

AI USE STATEMENT

In this work, no generative AI tools were used.

REPRODUCIBILITY STATEMENT

Data distributions, evaluation metrics, parameter selection rules, and baseline usage settings details are given in Section E. Model construction and objectives are in Sections A, B.1 and C. Additional results and diagnostics are in Sections F and G.

REFERENCES

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research*, 15(1):3563–3593, 2014.

Diana Cai, Chirag Modi, Charles Margossian, Robert Gower, David Blei, and Lawrence Saul. EigenVI: score-based variational inference with orthogonal function expansions. In *Advances in Neural Information Processing Systems*, 2024a.

Diana Cai, Chirag Modi, Loucas Pillaud-Vivien, Charles Margossian, Robert Gower, David Blei, and Lawrence Saul. Batch and match: black-box variational inference with a score-based divergence. In *International Conference on Machine Learning*, 2024b.

Diana Cai, Robert Gower, David Blei, and Lawrence Saul. Fisher meets Feynman: score-based variational inference with a product of experts. In *Advances in Neural Information Processing Systems*, 2025.

Chen-Hao Chao, Wei-Fang Sun, Bo-Wun Cheng, and Chun-Yi Lee. On investigating the conservative property of score-based generative models. *arXiv preprint arXiv:2209.12753*, 2022.

Taco S. Cohen and Max Welling. Group equivariant convolutional networks. In *International Conference on Machine Learning*, volume 48, pp. 2990–2999. PMLR, 2016.

Will Grathwohl, Kuan-Chieh Wang, Jörn-Henrik Jacobsen, David Duvenaud, Mohammad Norouzi, and Kevin Swersky. Your classifier is secretly an energy based model and you should treat it like one. In *International Conference on Learning Representations*, 2020.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, volume 32, pp. 15353–15363, 2019.

David J. Griffiths. *Introduction to Electrodynamics*. Pearson Education, 3 edition, 2005.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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

Aapo Hyvärinen. Some extensions of score matching. *Computational Statistics & Data Analysis*, 51(5):2499–2512, 2007.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, and Fu Jie Huang. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006.

Zengyi Li, Yubei Chen, and Friedrich T Sommer. Learning energy-based models in high-dimensional spaces with multiscale denoising-score matching. *Entropy*, 25(10):1367, 2023.

Cheng Lu, Kaiwen Zheng, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Maximum likelihood training for score-based diffusion ODEs by high order denoising score matching. In *International Conference on Machine Learning*, pp. 14429–14460. PMLR, 2022.

Benjamin Marlin, Kevin Swersky, Bo Chen, and Nando Freitas. Inductive principles for restricted Boltzmann machine learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 509–516, 2010.

Radford M Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

Tim Salimans and Jonathan Ho. Should EBMs model the energy or the score? In *ICLR Workshop on Energy Based Models*, 2021.

Hiroaki Sasaki, Aapo Hyvärinen, and Masashi Sugiyama. Clustering via mode seeking by direct estimation of the gradient of a log-density. In *Machine Learning and Knowledge Discovery in Databases, ECML PKDD 2014*, volume 8726 of *Lecture Notes in Computer Science*, pp. 19–34. Springer, 2014.

Hiroaki Sasaki, Takafumi Kanamori, Aapo Hyvärinen, Gang Niu, and Masashi Sugiyama. Mode-seeking clustering and density ridge estimation via direct estimation of density-derivative-ratios. *Journal of Machine Learning Research*, 18(180):1–47, 2018.

Harry Moritz Schey. *Div, Grad, Curl, and All That: An Informal Text on Vector Calculus*. W.W. Norton & Company, 3rd edition, 1996. ISBN 978-0393969979.

Jiaxin Shi, Shengyang Sun, and Jun Zhu. A spectral approach to gradient estimation for implicit distributions. In *International Conference on Machine Learning*, volume 80, pp. 4644–4653. PMLR, 2018.

Raghav Singhal, Mark Goldstein, and Rajesh Ranganath. What's the score? automated denoising score matching for nonlinear diffusions. *arXiv preprint arXiv:2407.07998*, 2024.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in Neural Information Processing Systems*, 33:12438–12448, 2020.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Conference on Uncertainty in Artificial Intelligence*, volume 115, pp. 574–584. PMLR, 2020.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34:1415–1428, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b.

Bharath K. Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Aapo Hyvärinen, and Revant Kumar. Density estimation in infinite dimensional exponential families. *Journal of Machine Learning Research*, 18(57):1–59, 2017.

Danica J. Sutherland, Heiko Strathmann, Michael Arbel, and Arthur Gretton. Efficient and principled score estimation with Nyström kernel exponential families. In *International Conference on Artificial Intelligence and Statistics*, volume 84, pp. 652–660. PMLR, 2018.

Kevin Swersky, Marc'Aurelio Ranzato, David Buchman, Nando D Freitas, and Benjamin M Marlin. On autoencoders and score matching for energy based models. In *International Conference on Machine Learning*, pp. 1201–1208, 2011.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011. doi: 10.1162/NECO_a_00142.

Li Wenliang, Danica J Sutherland, Heiko Strathmann, and Arthur Gretton. Learning deep kernels for exponential family densities. In *International Conference on Machine Learning*, pp. 6737–6746. PMLR, 2019.

Yilun Xu, Ziming Liu, Max Tegmark, and Tommi Jaakkola. Poisson flow generative models. *Advances in Neural Information Processing Systems*, 35:16782–16795, 2022.

Yilun Xu, Ziming Liu, Yonglong Tian, Shangyuan Tong, Max Tegmark, and Tommi Jaakkola. PFGM++: Unlocking the potential of physics-inspired generative models. In *International Conference on Machine Learning*, pp. 38566–38591. PMLR, 2023.

Yue Yang, Ryan Martin, and Howard Bondell. Variational approximations using Fisher divergence. *arXiv preprint arXiv:1905.05284*, 2019.

Longlin Yu and Cheng Zhang. Semi-implicit variational inference via score matching. In *International Conference on Learning Representations*, 2023.

Yuhao Zhou, Jiaxin Shi, and Jun Zhu. Nonparametric score estimators. In *International Conference on Machine Learning*, volume 119, pp. 11513–11522. PMLR, 2020.

A Electrostatic fields and potentials

We derive the fields in Sections 3.1 and 3.2 from their potentials. Throughout, the electric field and charge density associated with a potential V are

$$\boldsymbol{E}(\boldsymbol{x}) = -\nabla V(\boldsymbol{x}), \qquad q(\boldsymbol{x}) = \nabla \cdot \boldsymbol{E}(\boldsymbol{x}) = -\Delta V(\boldsymbol{x}). \tag{18}$$

These relationships hold in any dimension and for any geometry.

A.1 Spherically symmetric charge

Let $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{c}$ and $k_\sigma(\boldsymbol{x}, \boldsymbol{c}) = \exp(-\|\boldsymbol{r}\|^2/(2\sigma^2))$. The potential in eq. 11 is $V_{\text{sph}} = \sigma^2 k_\sigma$. Since

$$\nabla k_\sigma = -\frac{\boldsymbol{r}}{\sigma^2} k_\sigma, \qquad \Delta k_\sigma = \left(\frac{\|\boldsymbol{r}\|^2}{\sigma^4} - \frac{d}{\sigma^2} \right) k_\sigma,$$

then eq. 18 gives

$$\boldsymbol{E}_{\text{sph}}(\boldsymbol{x}) = \boldsymbol{r} k_\sigma(\boldsymbol{x}, \boldsymbol{c}), \qquad q_{\text{sph}}(\boldsymbol{x}) = \left(d - \frac{\|\boldsymbol{r}\|^2}{\sigma^2} \right) k_\sigma(\boldsymbol{x}, \boldsymbol{c}). \tag{19}$$

Thus, the charge changes sign at $\|\boldsymbol{r}\| = \sigma\sqrt{d}$. Its integral over the space $\mathbb{R}^d$ is zero; by Gauss's law, this integral equals the outward flux of $\boldsymbol{E}_{\text{sph}}$ at infinity, which then vanishes because the field decays exponentially. The field and charge are therefore localized, although neither has a finite spherical boundary.

A.2 Finite slabs and one-sided boundaries

Let $\boldsymbol{u}$ be a unit vector. As parameterized in Section 3.2, for a slab of half-width L and uniform charge density q_1, the potential can be written as

$$V_{\text{slab}}(\boldsymbol{x}) = \begin{cases} -\boldsymbol{u}^\top \boldsymbol{x} - b^2/2, & \text{if } |\boldsymbol{u}^\top \boldsymbol{x} - b| \le L, \\ -L|\boldsymbol{u}^\top \boldsymbol{x} - b| + L^2/2, & \text{if } |\boldsymbol{u}^\top \boldsymbol{x} - b| > L, \end{cases} \tag{20}$$

The two cases and their first derivatives are equal at $\boldsymbol{u}^\top \boldsymbol{x} - b = \pm L$. As $\nabla(\boldsymbol{u}^\top \boldsymbol{x} - b) = \boldsymbol{u}$, differentiating the potential yields the field:

$$\boldsymbol{E}_{\text{slab}}(\boldsymbol{x}) = \begin{cases} q_1(\boldsymbol{u}^\top \boldsymbol{x} - b)\,\boldsymbol{u} & \text{if } |\boldsymbol{u}^\top \boldsymbol{x} - b| \le L \\ q_1 L\,\boldsymbol{u} & \text{if } \boldsymbol{u}^\top \boldsymbol{x} - b > L \\ -q_1 L\,\boldsymbol{u} & \text{if } \boldsymbol{u}^\top \boldsymbol{x} - b < -L, \end{cases}$$

These are the field and charge stated in Eq. 14.

The fitted slab dictionary uses two one-sided components in Eq. 15. To formalize their equivalence to the finite slab, consider boundaries placed at $b - L$ and $b + L$, with the same normal $\boldsymbol{u}$:

$$\begin{aligned} \boldsymbol{E}_{\text{slab}}(\boldsymbol{x}) &= q_1 \left([\boldsymbol{u}^\top \boldsymbol{x} - b + L]_+ - [\boldsymbol{u}^\top \boldsymbol{x} - b - L]_+ - L \right) \boldsymbol{u}, \\ q_{\text{slab}}(\boldsymbol{x}) &= q_1 \left(\mathbf{1}\{\boldsymbol{u}^\top \boldsymbol{x} - b > -L\} - \mathbf{1}\{\boldsymbol{u}^\top \boldsymbol{x} - b > L\} \right). \end{aligned} \tag{21}$$

The potential is therefore $-\frac{q_1}{2}[\boldsymbol{u}^\top \boldsymbol{x} - b + L]_+^2 + \frac{q_1}{2}[\boldsymbol{u}^\top \boldsymbol{x} - b - L]_+^2 + q_1 L(\boldsymbol{u}^\top \boldsymbol{x} - b)$, up to an additive constant. And thus, a pair of one-sided components with a constant field $-q_1 L \boldsymbol{u}$ reproduce the finite slab exactly. Conversely, an individual fitted boundary has a charge on a half-space, and by itself does not adequately compose a finite slab.

B Training and optimization

This section gives the conditional updates used to fit a fixed charge dictionary. The empirical objective is eq. 3, and ridge regularization is applied to the charge coefficients as in Section 3.4. Construction of geometric bases are specified separately in Section C.

B.1 Optimization over learned coefficients

We derive the quadratic form of the empirical Fisher divergence (eq. 3) under the score model (eq. 16). Substituting $\boldsymbol{s}_\theta(\boldsymbol{x}) = \sum_{k=1}^K \theta_k \boldsymbol{\phi}_k(\boldsymbol{x}) + \boldsymbol{\Omega}(\boldsymbol{x} - \boldsymbol{x}_0)$ into the empirical objective, the squared norm term expands as

$$\begin{aligned} \|\boldsymbol{s}_\theta(\boldsymbol{x})\|^2 &= \left\| \sum_{k=1}^K \theta_k \boldsymbol{\phi}_k(\boldsymbol{x}) + \boldsymbol{\Omega}(\boldsymbol{x} - \boldsymbol{x}_0) \right\|^2 \\ &= \sum_{k,l=1}^K \theta_k \theta_l \boldsymbol{\phi}_k(\boldsymbol{x})^\top \boldsymbol{\phi}_l(\boldsymbol{x}) \\ &\quad + 2 \sum_{k=1}^K \theta_k \boldsymbol{\phi}_k(\boldsymbol{x})^\top \boldsymbol{\Omega}(\boldsymbol{x} - \boldsymbol{x}_0) \\ &\quad + \|\boldsymbol{\Omega}(\boldsymbol{x} - \boldsymbol{x}_0)\|^2. \end{aligned} \tag{22}$$

The divergence term is

$$\nabla \cdot \boldsymbol{s}_\theta(\boldsymbol{x}) = \sum_{k=1}^K \theta_k \rho_k(\boldsymbol{x}) + \operatorname{tr}(\boldsymbol{\Omega}), \tag{23}$$

where we have that $\rho_k(\boldsymbol{x}) = \nabla \cdot \boldsymbol{\phi}_k(\boldsymbol{x})$ is the charge density of component k, as in eq. 16.

Substituting eq. 22 and eq. 23 into the empirical objective

$$\widehat{\mathcal{J}}(\boldsymbol{s}_\theta) = \frac{1}{n}\sum_{i=1}^{n}\left(\|\boldsymbol{s}_\theta(\boldsymbol{x}_i)\|^2 + 2\nabla\cdot\boldsymbol{s}_\theta(\boldsymbol{x}_i)\right),$$

the terms quadratic in θ are

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{k,l}\theta_k\theta_l\boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top\boldsymbol{\phi}_l(\boldsymbol{x}_i) = \sum_{k,l}\theta_k\theta_l G_{kl},$$

and the terms linear in λ are

$$\frac{2}{n}\sum_{i=1}^{n}\sum_{k}\theta_k\left[\boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top\boldsymbol{\Omega}(\boldsymbol{x}_i-\boldsymbol{x}_0)+\rho_k(\boldsymbol{x}_i)\right] = 2\sum_k\theta_k g_k.$$

where

$$G_{kl} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top\boldsymbol{\phi}_l(\boldsymbol{x}_i),$$
$$g_k = \frac{1}{n}\sum_{i=1}^{n}\left(\boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top\boldsymbol{\Omega}(\boldsymbol{x}_i-\boldsymbol{x}_0)+\nabla\cdot\boldsymbol{\phi}_k(\boldsymbol{x}_i)\right).$$

The remaining terms involving only $\boldsymbol{\Omega}$ are constant with respect to λ, giving the quadratic in eq. 17.

B.2 Optimization over $\boldsymbol{\Omega}, \mathbf{x}_0$

Treating $\boldsymbol{\Omega}$ as a free parameter and collecting the terms involving Ω,

$$\widehat{\mathcal{J}}(\lambda,\boldsymbol{\Omega}) = \frac{1}{n}\sum_{i=1}^{n}\left[\|\boldsymbol{\Omega}(\boldsymbol{x}_i-\boldsymbol{x}_0)\|^2 + 2\sum_k\theta_k\boldsymbol{\phi}_k(\boldsymbol{x}_i)^\top\boldsymbol{\Omega}(\boldsymbol{x}_i-\boldsymbol{x}_0) + 2\,\mathrm{tr}(\boldsymbol{\Omega})\right] + \text{const. w.r.t. } \boldsymbol{\Omega}.$$

Define the centered $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i - \boldsymbol{x}_0$ as well as the sample covariance $\mathbf{C} = \frac{1}{n}\sum_{i=1}^{n}\tilde{\boldsymbol{x}}_i\tilde{\boldsymbol{x}}_i^\top$. Differentiating with respect to $\boldsymbol{\Omega}$ (treating it as symmetric, so the gradient is symmetrized):

$$\frac{\partial\widehat{\mathcal{J}}}{\partial\boldsymbol{\Omega}} = \boldsymbol{\Omega}\mathbf{C} + \mathbf{C}\boldsymbol{\Omega} + \mathbf{R} + \mathbf{R}^\top + 2\mathbf{I} = 0,$$

where $\mathbf{R} = \frac{1}{n}\sum_{i=1}^{n}\sum_k\theta_k\boldsymbol{\phi}_k(\boldsymbol{x}_i)\tilde{\boldsymbol{x}}_i^\top$. This simplifies to the Lyapunov equation

$$\boldsymbol{\Omega}\mathbf{C} + \mathbf{C}\boldsymbol{\Omega} = -(\mathbf{R}+\mathbf{R}^\top) - 2\mathbf{I}. \tag{24}$$

The shift parameter x_0 also appears in the objective through the linear field $\boldsymbol{\Omega}(\boldsymbol{x}-\boldsymbol{x}_0)$. Differentiating $\hat{\mathcal{J}}$ w.r.t. $\boldsymbol{x}_0$ and using the symmetry of $\boldsymbol{\Omega}$,

$$\frac{\partial\widehat{\mathcal{J}}}{\partial\boldsymbol{x}_0} = -\frac{2}{n}\sum_{i=1}^{n}\boldsymbol{\Omega}\left[\boldsymbol{\Omega}(\boldsymbol{x}_i-\boldsymbol{x}_0) + \sum_{k=1}^{K}\theta_k\boldsymbol{\phi}_k(\boldsymbol{x}_i)\right] = 0.$$

Setting this to zero and solving (assuming $\boldsymbol{\Omega}$ is invertible),

$$\boldsymbol{x}_0^\star = \bar{\boldsymbol{x}} + \boldsymbol{\Omega}^{-1}\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K}\theta_k\,\boldsymbol{\phi}_k(\boldsymbol{x}_i), \tag{25}$$

where $\bar{\boldsymbol{x}} = \frac{1}{n}\sum_i\boldsymbol{x}_i$ is the sample mean. In our experiments, $\boldsymbol{x}$ is initialized at the sample mean and updated together with $\boldsymbol{\Omega}$ during joint refitting using eqs. 24 and 25. This is the linear Gaussian baseline. In our experiments, $\boldsymbol{x}_0$ is initialized to $\bar{\boldsymbol{x}}$ and then updated jointly with $\boldsymbol{\Omega}$ at each Lyapunov refinement step using eq. 25.

B.3 Joint refitting procedure

We use the same joint Lyapunov fitting procedure in all experiments. For a fixed dictionary, the geometric coefficients and symmetric liner base field are fit together under the regularized score matching objective, eq. 3. The coupled fit is described by the coefficient equations in Section B.1, the Lyapunov equation in eq. 24, and the center condition in eq. 25.

Although the model and coefficient-fitting procedure are shared across experiments, the geometric dictionaries and their selection differ: synthetic experiments use validation to select among candidate fits, whereas WAE training instead uses a fixed component budget and constructs spherical centers from the given latent minibatch, since there is no separate validation search within each update; see Section E.6.2.

B.4 Numerical considerations and complexity

For n training samples, dimension d, and K components in the selected charge dictionary, forming the Gram matrix in eq. 17 requires $O(ndK^2)$ operations and $O(K^2)$ storage. Since the component geometries remain fixed during refitting, this matrix can be computed only once for a given dictionary. A direct solution of the regularized $K \times K$ system costs $O(K^3)$.

The linear update in eq. 24 is a $d \times d$ Lyapunov solve and costs $O(d^3)$ with standard dense solver. The $\boldsymbol{x}_0$ update needs only the fit component fields and the linear solve in $\boldsymbol{\Omega}$. These costs are comparatively small for the adaptive model with $K \ll n$.

We add ridge regularization to the model to improve conditioning, detailed further in Section 3.4. As is typical for score matching models and applied as well to our baseline comparisons, we use an independent validation set for hyperparameter selection and the stopping rule; full details are in Section C. WAE fits use fixed settings within each update, as described in Section E.6.2.

C Dictionary construction and adaptive fitting

A dictionary specifies the component fields before their final coefficients are fit. For a spherical component, we choose a center and width; for a slab boundary, the unit normal vector and breakpoint. These choices determine both the field and its divergence through Eqs. 15 and 19. We then fit the coefficients by the quadratic solve in Section B.3. The synthetic dictionary search below uses observations and validation losses. WAE dictionary construction uses the fixed minibatch budgets in Section E.6.2; both use the same joint coefficient fit.

C.1 Spherical centers and slab directions

In two dimensions, every fitting observation is a candidate spherical center. Both dense and adaptive models therefore draw from the same $1,600$ centers. Each fitted dictionary uses a common spherical width, selected from the grid in Section E.3. The dimension sweep uses 128 k-means centers instead as a possible dictionary to choose from; settings in Section E.4.1.

For slabs, we begin with k directions from FastICA, using at most 500 iterations and the experiment seed for FastICA, and normalize each direction to unit length. Then along each direction, place breakpoints based on Gaussian quantiles within the range along each such direction, clipped to $[0.001, 0.999]$. We sample one offset uniformly within each bin. The number of breakpoints per direction is chosen from $\{4, 5, 7\}$ by validation of the initial ICA model.

We then seek directions in which the current fit score fails the Stein identity:

$$\mathbb{E}_p[f(\boldsymbol{x})\nabla \log p(\boldsymbol{x}) + \nabla f(\boldsymbol{x})] = \mathbf{0}, \tag{26}$$

which holds for smooth test function f. Replacing the true score with our current estimate give an empirical residual that measures its violate of this identity. We use this to select additional slab directions.

For regularization, we test ℓ_2 ridge strengths $\{0.01, 0.05, 0.1, 0.5\}$ and fit the coefficients and linear base field with the joint Lyapunov procedure before computing validation loss. We retain the best

dictionary seen, stopping after five steps without improvement, or at 25 addition steps, whichever occurs first.

For a full slab dictionary, we retain the selected $\{\boldsymbol{u}\}, \{b\}$ and fill the remaining budget to $1,600$. We cycle over the distinct recovered directions and place additional offsets at evenly spaced midpoint quantiles of their training projections.

C.2 Full dictionaries and adaptive subsets

A full-dictionary model fits every component in its dictionary: 1,600 spheres, 1,600 slab boundaries, or their 3,200-component union. Adaptive fitting instead constructs a sequence of smaller dictionaries. Forward selection adds the component of this dictionary that most reduces the regularized training objective after accounting for the existing components and the fitted affine field. The coefficients are then refit together. This ranking uses the quadratic objective.

D Connection to existing score estimators

The empirical score-matching objective in Eq. 3 evaluates $\|\boldsymbol{s}(\boldsymbol{x}_i)\|^2$ and $\nabla \cdot \boldsymbol{s}(\boldsymbol{x}_i)$ at each observation. Methods differ in how they define $\boldsymbol{s}$, how they obtain its divergence, and which parameters they fit. We discuss these choices for neural and kernel estimators, then relate them to the charge model in Eq. 16.

D.1 Neural score estimators

D.1.1 Direct-score neural estimators

A direct neural score estimator maps $\boldsymbol{x} \in \mathbb{R}^d$ to a vector $\boldsymbol{s}_\eta(\boldsymbol{x}) \in \mathbb{R}^d$. The network output supplies the first term of Eq. 3. The second term requires the trace of the network's input Jacobian:

$$\nabla \cdot \boldsymbol{s}_\eta(\boldsymbol{x}) = \sum_{j=1}^{d} \frac{\partial s_{\eta,j}(\boldsymbol{x})}{\partial x_j}. \tag{27}$$

These input derivatives can be computed by differentiation. A trace requires its diagonal terms, and training the network weights differentiates the resulting objective with respect to η. Direct neural score models are used in score-based generative modeling and in neural score estimation with sliced score matching (Song & Ermon, 2019; Song et al., 2020; 2021b).

These networks do not necessarily have a symmetric Jacobian, $\frac{\partial s_{\eta,j}}{\partial x_\ell} \neq \frac{\partial s_{\eta,\ell}}{\partial x_j}$. This has been studied for different learned score models in Chao et al. (2022). It concerns the model parameterization, as minimizing the score-matching objective does not impose Jacobian symmetry, and therefore conservativeness.

Sliced Score Matching (SSM) reduces the cost of the derivative term. For random directions with $\mathbb{E}(\boldsymbol{v}\boldsymbol{v}^\top) = \mathbf{I}$, SSM constructs $\nabla \cdot \boldsymbol{s}_\eta(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{v}}\left[\boldsymbol{v}^\top(\nabla \boldsymbol{s}_\eta(\boldsymbol{x}))\boldsymbol{v}\right]$. This expectation is estimated using a finite number of directions and vector products (Song et al., 2020). It avoids using an exact trace, although it is impacted by the sampling variance of the directions. SSM does not itself make a direct score network conservative.

D.1.2 Scalar potential neural estimators

A second approach makes the network output a scalar log-density $F_\eta(\boldsymbol{x})$, up to an additive constant. The modeled score and its divergence are

$$\boldsymbol{s}_\eta(\boldsymbol{x}) = \nabla F_\eta(\boldsymbol{x}), \qquad \nabla \cdot \boldsymbol{s}_\eta(\boldsymbol{x}) = \Delta F_\eta(\boldsymbol{x}). \tag{28}$$

Thus, evaluating eq. 3 requires the gradient and the Laplacian of the scalar network: $\|\nabla F_\eta(\boldsymbol{x}_i)\|^2 + 2\Delta F_\eta(\boldsymbol{x}_i)$.

The score is conservative by construction. The Laplacian is the trace of the input Hessian, thus training differentiates that trace with respect to the network parameters.

Sliced score matching can also be applied to this scalar parameterization, replacing the Laplacian by an average of Hessian quadratic forms along random directions. This reduces the cost of evaluating the trace while preserving conservativeness. Training still must then differentiate these input derivatives with respect to the network weights (Song et al., 2020).

Neural energy and direct score methods are also compared in Salimans & Ho (2021).

Our model also begins with scalar potentials, but it chooses components with fields and Laplacians that are given explicitly, thus once geometries are fixed, evaluating the score matching objective eq. 3 doesn't require the inefficient differentiation through a neural network.

D.2 Gaussian kernels and spherical charge

A finite Gaussian kernel exponential family (KEF) uses the unnormalized log-density

$$\log \widetilde{p}(\boldsymbol{x}) = \log p_0(\boldsymbol{x}) + \sum_{m=1}^{M} a_m k_{\sigma_m}(\boldsymbol{x}, \boldsymbol{c}_m), \qquad k_\sigma(\boldsymbol{x}, \boldsymbol{c}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{2\sigma^2}\right). \tag{29}$$

The score matching objective needs the gradient and Laplacian of each kernel section. As in our electrostatic model, the are analytic in nature,

$$\nabla k_\sigma(\boldsymbol{x}, \boldsymbol{c}) = -\frac{\boldsymbol{E}_{\text{sph}}(\boldsymbol{x})}{\sigma^2}, \qquad \Delta k_\sigma(\boldsymbol{x}, \boldsymbol{c}) = -\frac{q_{\text{sph}}(\boldsymbol{x})}{\sigma^2}, \tag{30}$$

where the field and charge are given in eq. 11. Therefore,

$$\nabla \log \widetilde{p}(\boldsymbol{x}) = \nabla \log p_0(\boldsymbol{x}) - \sum_m \frac{a_m}{\sigma_m^2} \boldsymbol{E}_{\text{sph},m}(\boldsymbol{x}), \tag{31}$$

$$\Delta \log \widetilde{p}(\boldsymbol{x}) = \Delta \log p_0(\boldsymbol{x}) - \sum_m \frac{a_m}{\sigma_m^2} q_{\text{sph},m}(\boldsymbol{x}). \tag{32}$$

Both terms of the empirical objective are therefore available directly. With fixed centers and widths, the objective is quadratic in $\boldsymbol{\alpha}$. If p_0 in the above expressions is Gaussian, its score is equivalent to our linear base term. Thus with matched centers, σ, and base score, Eq. 29 is exactly the spherical-only score family in Section 3.1. For the fields as written in Eq. 11, the coefficients relate as $\theta_m = -a_m/\sigma_m^2$. This is changed by feature normalization and does not impact the learned score. Thus, Gaussian KEF models provide an existing example of the spherical charge model, and the slab components extend that finite dictionary beyond the KEF parametrization.

An alternative form of regularization that we explore in Section G is the common kernel exponential family RKHS regularization. It is an RKHS penalty on $f = \sum_m a_m k_\sigma(\cdot, \boldsymbol{c}_m)$ and is written as

$$\|f\|_{\mathcal{H}_k}^2 = \boldsymbol{a}^\top \mathbf{K}_{CC} \boldsymbol{a}, \qquad (\mathbf{K}_{CC})_{ml} = k_\sigma(\boldsymbol{c}_m, \boldsymbol{c}_l),$$

whereas Section 3.4 penalizes charge coefficients with ℓ_2 regularization: $\alpha\|\boldsymbol{\theta}\|^2$. Different regularizers or centers can give different fitted estimators even when the spherical densities are constructed the same.

D.2.1 Kernel basis points and KEF-CG

The finite scalar expansion above assigns coefficients to kernel sections centered at selected basis points, however there are several different methods that kernel score estimators use. Fro example, they may use derivative representers, whose vector-valued basis functions are obtained from derivatives of a scalar kernel (Zhou et al., 2020). Their coefficients also follow from a regularized linear system, the same as our electrostatic score model, and its spherical-component-only equivalent, Gaussian KEF. However, the basis of these KEFs and penalty are not necessarily equivalent to the finite Gaussian KEF. A curl-free kernel can enforce conservativeness directly.

A limitation of full-basis kernel methods is their computational complexity. Conjugate gradients sovle the coefficient system through matrix–vector products, without needing to invert its matrix. This therefore changes the numerical solver, not the statistical objective. Therefore, reproducing a KEF-CG comparison requires specifying the kernel, representer construction, solver tolerance, and regularization. The experimental choices are given in Section E.5.

D.2.2 Deep kernel exponential families

A deep kernel exponential family (DKEF) learns a neural feature map inside the kernel (Wenliang et al., 2019). Its linear coefficients are fitted by a regularized score matching loss, conditioned on the neural feature map and its inducing points. The feature map and inducing points are then optimized by gradient-based training. In the implementation used here, separate training batches supply the inner coefficient fit and the outer training objective.

In DKEF, the score remains the gradient of a scalar log-density, but evaluating the score-matching loss requires input derivatives of the learned kernel. As with the score-matching objective, training the feature map requires differentiating through those quantities and the coefficient solve. This differs from a fixed electrostatic dictionary, where the field and divergence are explicit and only the final coefficients are optimized. The network and training settings are listed in Section E.5.

D.3 Slab fields as tied-direction ReLU networks

As stated in Section 3.2, a one-sided slab field is $\phi_{\mathbf{u},b}(\mathbf{x}) = [\mathbf{u}^\top \mathbf{x} - b]_+ \mathbf{u}$. It is the gradient of $\psi_{\mathbf{u},b}(\mathbf{x}) = \frac{1}{2}[\mathbf{u}^\top \mathbf{x} - b]_+^2$. The slab model alone therefore has the form

$$\mathbf{s}(\mathbf{x}) = \mathbf{\Omega}(\mathbf{x} - \mathbf{x}_0) + \sum_{k=1}^{K} \theta_k [\mathbf{u}_k^\top \mathbf{x} - b_k]_+ \mathbf{u}_k, \tag{33}$$

which is a one-hidden-layer ReLU network with tied input and output directions, as pictured in Figure 6. For comparison, a general untied ReLU unit has the form $a_k[\mathbf{w}_k^\top \mathbf{x} - b_k]_+ \mathbf{v}_k$. Away from the activation boundary, this ReLU unit contributes $\theta_k \mathbf{1}\{\mathbf{w}_k^\top \mathbf{x} > b_k\} \mathbf{v}_k \mathbf{w}_k^\top$ to the Jacobian, which is not generally symmetric for $\boldsymbol{v}_k \neq \boldsymbol{w}_k$. It can be rewritten trivially as our tied ReLU unit in the slab model when $\boldsymbol{v}_k = \boldsymbol{w}_k = \boldsymbol{u}_k$ relationship is enforced. Then the contribution to the Jacobian is the symmetric $\theta_k \mathbf{1}\{\mathbf{u}_k^\top \mathbf{x} > b_k\} \mathbf{u}_k \mathbf{u}_k^\top$. Our slab charge model, with the tied-direction equivalence, is therefore conservative by construction, whereas the untied ReLU network generally is not.

The tied-direction model is also analytic:

$$\nabla \cdot \left(\theta_k [\mathbf{u}_k^\top \mathbf{x} - b_k]_+ \mathbf{u}_k\right) \theta_k \|\mathbf{u}_k\|^2 \mathbf{1}\{\mathbf{u}_k^\top \mathbf{x} > b_k\}. \tag{34}$$

For unit $\boldsymbol{u}_k$, this reduces to the one-sided charge slab density. Thus, both the score and divergence are availale directly. Neural score models can use more flexible parameterizations, but are generally fitted by iterative optimization. For a fixed electrostatic dictionary, the coefficients instead follow from the quadratic system in Section 3.4.

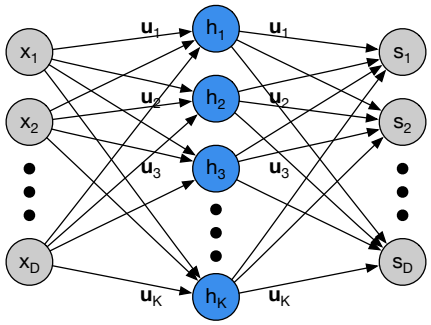


Figure 6: Neural network implementation of the charged slab electrostatic score model. A single hidden layer of arbitrary width consists of a mixture of ReLU and linear neurons. The input weights $\boldsymbol{u}_k$ to the k-th hidden neuron are tied to the output weights $\boldsymbol{u}_k$ of that neuron, enforcing curl-free structure. Untying these weights recovers a standard MLP.

D.4 Relationship to least-squares log-density gradients

LSLDG (Sasaki et al., 2014; 2018) uses Gaussian-derivative features centered at data points $\boldsymbol{c}_i$,

$$g_j(\boldsymbol{x}) = \sum_i \theta_{i,j}\, \partial_j \phi_i(\boldsymbol{x})$$
$$\phi_i(\boldsymbol{x}) = \exp\big(-\|\boldsymbol{x} - \boldsymbol{c}_i\|^2 / 2\sigma^2\big),$$

which is quadratic in the coefficients and solved in closed form, exactly the structure of our eq. 17. The per-coordinate LSLDG estimator is not curl-free in general: with independent coefficients across coordinates, $(\nabla \times g)_{j\ell} = \sum_i (\theta_{i,j} - \theta_{i,\ell})\, \partial_j \partial_\ell \phi_i \neq 0$. It should be noted that only by tying coefficients across coordinates, $\theta_{i,j} = \theta_i$, gives $g = \nabla(\sum_i \theta_i \phi_i)$, which is curl-free.

D.5 Other score estimators

The spectral Stein gradient estimator (SSGE) (Shi et al., 2018) estimates each score coordinate through a spectral expansion, where the basis functions approximate eigenfunctions of a kernel integral operator. The coefficients follow from Stein's identiy, not the Fisher-divergence objective. In its standard form in Shi et al. (2018), SSGE estimates separate coefficients for each output coordinate and therefore does not enforce a shared scalar potential. For M samples in dimension d, Shi et al. (2018) the construction cost is dominated by the kernel eigendecomposition and Gram matrix.

E Experimental protocol

E.1 Data distributions

The two-dimensional experiments use Banana, GMM, and X-shape distributions. The Gaussian density can be written as $\varphi_d(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. True scores $\nabla \log p(\boldsymbol{x})$ are used only for evaluation, and model selection uses the empirical score-matching objective in Eq. 3. Higher-dimensional extensions are defined in Section F.3.

E.1.1 Banana

For independent $Z_1, Z_2 \sim \mathcal{N}(0, 1)$, let

$$X_1 = Z_1, \qquad X_2 = Z_2 + \frac{Z_1^2 - 1}{2}. \tag{35}$$

The transformation has unit Jacobian determinant, giving

$$p_{\mathrm{B}}^{(2)}(\boldsymbol{x}) = \frac{1}{2\pi} \exp\left[-\frac{1}{2}\left\{x_1^2 + \left(x_2 - \frac{x_1^2 - 1}{2}\right)^2\right\}\right]. \tag{36}$$

E.1.2 GMM

We use four equally weighted Gaussians whose centers form a square at radius 5 from the origin and with identity covariance:

$$p_{\mathrm{G}}^{(2)}(\boldsymbol{x}) = \frac{1}{4} \sum_{a,b \in \{-1,+1\}} \varphi_2\left(\boldsymbol{x}; \frac{5}{\sqrt{2}}(a, b)^\top, \mathbf{I}_2\right). \tag{37}$$

E.1.3 X-shape

We use two equally weighted Gaussians with the same mean, $\boldsymbol{\mu} = (1, 1)^\top$:

$$p_{\mathrm{X}}^{(2)}(\boldsymbol{x}) = \frac{1}{2} \sum_{s \in \{-1,+1\}} \varphi_2(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}_s), \qquad \boldsymbol{\Sigma}_s = \begin{pmatrix} 1 & 0.95s \\ 0.95s & 1 \end{pmatrix}. \tag{38}$$

E.2 Data splits and evaluation

The two-dimensional comparison uses $1,600$ fitting observations, 400 validation observations, and $5,000$ independent test observations. The dimension sweep uses $1,600$ fitting, 400 validation, and $5,000$ test observations at each $d \in \{5, 10, 15, 30, 50, 75, 100\}$. Each main comparison contains five runs at five different seeds. Methods within a each experiment section use the same data-generation and splitting protocol.

The seeds are 5101, 5202, 5303, 5404, 5505. For each seed, we draw $2,000$ observations with NumPy's seed range, and retain the first 1,600 as the fitting set T, the next 400 as validation set V. We then use a 5,000 draw for test observations. For fitting, the combined set is exactly $T \cup V$.

We estimate the Fisher divergence (Eq. 1) defined in Section 2.1, by the test score error:

$$\widehat{\mathrm{MSE}} = \frac{1}{n_{\mathrm{test}}} \sum_{i=1}^{n_{\mathrm{test}}} \|\widehat{\boldsymbol{s}}(\boldsymbol{x}_i) - \boldsymbol{s}^\star(\boldsymbol{x}_i)\|^2. \tag{39}$$

For all dimension comparisons, this is total squared error without division by d. Tables report the mean and one standard error across seeds. The analytic scores obtained from the densities in Eqs. 36 to 38 are used only for evaluation.

E.3 Electrostatic models and selection

The experiments compare the spherical fields in Section 3.1, written as Eq. 11, and the slab fields in Section 3.2. The mixed model includes both of these families. Additional details in Section A. The models use the dictionaries in Section C. All models include a symmetric linear base field, as in Eq. 16. For fixed geometry, the coefficients solve the regularized quadratic in Eq. 17. The linear field is solved using the symmetric Lyapunov solution in Section B.2. The resulting solve jointly fits the linear field and the geometric coefficients. Each adaptive checkpoint is a joint fit, and earlier coefficients are not frozen. We define and use throughout K as the number of geometric coefficients (or components), excluding parameters in the linear base.

E.3.1 Validation loss

Validation estimates the same score-matching objective used for fitting. For convenience, the implementation uses half of eq. 3:

$$J_A(\boldsymbol{s}) = \frac{1}{|A|} \sum_{\boldsymbol{x} \in A} \left[\frac{1}{2} \|\boldsymbol{s}(\boldsymbol{x})\|^2 + \nabla \cdot \boldsymbol{s}(\boldsymbol{x}) \right]. \tag{40}$$

Training then minimizes $J_T + \boldsymbol{\theta}^\top \mathbf{R} \boldsymbol{\theta}/2$; validation evaluates J_V without the coefficient penalty. The analytic score is not needed for either quantity.

We consider and use two adjustments to plain validation loss. The first also fits the coefficients on V, keeping the geometry and the training-defined penalty matrix fixed. However, it adds terms to account for how much the fitted objective changes when the fitting and validation sets exchange roles. The reverse fit is not used in the training fit, it is only a method of validation selection. Writing these two fits as $\widehat{\boldsymbol{s}}_T$ and $\widehat{\boldsymbol{s}}_V$, the criterion is

$$C_{\mathrm{sym}} = J_V(\widehat{\boldsymbol{s}}_T) + 0.1\big[J_T(\widehat{\boldsymbol{s}}_V) - J_T(\widehat{\boldsymbol{s}}_T) + J_V(\widehat{\boldsymbol{s}}_T) - J_V(\widehat{\boldsymbol{s}}_V)\big]. \tag{41}$$

The second validation adjustment we use is the standard $\pm$SE adjustment,

$$C_c = J_V(\widehat{\boldsymbol{s}}_T) + c\,\widehat{\mathrm{SE}}_V, \tag{42}$$

where $\widehat{\mathrm{SE}}_V$ is the sample standard deviation of the per-observation validation loss. This favors candidates with lower mean loss and less variable validation loss. The fixed multiplier c is just a selection heuristic, we do not calibrate it. The sphere-based models minimize C_{sym}, and the slab-based models minimize C_1. The adaptive mixed model minimizes over $C_{1.5}$. In the reported ablations in Section F.2, its full-dictionary counterpart uses C_{sym}.

E.4 Widths and regularization in two dimensions

Slab models use ℓ_2 regularization. Full-dictionary and adaptive slab-only fits search strengths $\{10^{-4}, 10^{-2}, 10^{-1}, 1\}$.

Spherical models search widths $\{0.35, 0.7, 1.4, 2.8\}$ and regularization strengths $\{10^{-4}, 10^{-2}, 10^{-1}, 1\}$. Full-dictionary spherical models use feature-normalized ℓ_2 regularization. Adaptive spherical models consider both this penalty and Gaussian RKHS regularization (Section D.2).

Mixed models regularize the two families separately: adaptive mixed models use Gaussian RKHS regularization for spheres and ℓ_2 for slabs, and search spherical strengths $\{10^{-4}, 10^{-2}, 10^{-1}, 1\}$ and slab strengths equal to $\{0.1, 1, 10\}$ times the spherical strength. In the full-dictionary ablation, Section F.2, the full-dictionary mixed models consider both spherical penalties above, with ℓ_2 for slabs, and independently search

$$\alpha_{\text{sph}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}, \qquad \alpha_{\text{slab}} \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}.$$

Validation selects among these with the criteria described in Section E.3.1.

E.4.1 Dimension-sweep settings

The full-rank sweep uses the same slab-direction construction of ICA + Stein, but a smaller spherical proposal pool: 128 k-means centers, one initialization at the experiment seed.

E.5 Baseline implementations

We implement both finite scalar Gaussian kernel models, which use the log-density expansion in Section D.2, derivative-representer kernel estimators, and the SSM neural method. See Section D.2 for more details on the different forms of the KEF models, and Section D.1 for SSM.

SSGE (Shi et al., 2018) uses the spectral-cutoff estimator with a Gaussian kernel. Gaussian KEF-CG and Curl-free IMQ use the derivative-representer estimators of Zhou et al. (2020). For these kernel comparisons, we use width multipliers as $\{0.5, 1, 1.5, 2, 3\}$ times the training pair-distance scale, and regularization strengths, or spectral cutoffs for SSGE, are $\{10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1\}$. Selection minimizes the validation loss in eq. 40.

SSM (Song et al., 2020) uses a direct score network with two Softplus hidden layers of width 128 and four Gaussian projection vectors per observation. Adam learning rates are $\{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$. Training runs for at most 1,200 steps, with validation every 50 steps and a patience of 400 steps. The model details of this method are described in Section D.1.1.

DKEF (Wenliang et al., 2019) uses the public PyTorch implementation distributed with the SSM code: 64 learned inducing points, hidden width 30, and its native Gaussian base. It trains for at most 600 steps, checking validation every 30 steps and with patience of 180 steps. These two-dimensional baselines are do not contain the additional fitted electrostatic linear base field in the other kernel models and our electrostatic models. The DKEF model was too computationally intensive to implement for any higher-dimensional datasets, such as the synthetic dimension sweep or the WAE on MNIST or CIFAR-10.

In the dimension sweep, the finite Gaussian KEF (Sriperumbudur et al., 2017; Sutherland et al., 2018) uses all 1,600 fitting observations as centers, with a 256-center variant included in the kernel diagnostic. It fits a Gaussian base once and holds that base fixed. Width multipliers are $\{0.125, 0.25, 0.5, 1, 2, 4\}$ and RKHS strengths range from 10^{-8} to 1. The completed Gaussian derivative KEF-CG and SSGE comparisons use the five-width, five-regularization grids given above, on all five runs at each dimension.

E.6 Wasserstein autoencoder experiments

The WAE experiments use the same electrostatic score model and joint Lyapunov fit as the synthetic experiments. At every training update, we construct the dictionary from the current encoder latent dimension samples and jointly fit its coefficients and linear base field. Because there is no separate

validation set for selecting a score model within each minibatch, we fix the component budget and fitting settings across models, and the spherical centers are chosen from each given minibatch.

E.6.1 Data, networks, and training

We adapt the WAE implementation of Song et al. (2020). For MNIST, we use their deterministic encoder and decoder, each with two 256-unit Tanh hidden layers and a sigmoid decoder output. For CIFAR-10, we adapt their convolutional architecture to 32×32 RGB images, using an $512 \times 2 \times 2$ initial decoder feature map and a sigmoid output. The architecture and autoencoder training settings are shared across all tested score estimators for fair comparisons.

MNIST uses 48,000 training and 12,000 diagnostic images, obtained by shuffling the standard training set with seed 2019. CIFAR-10 uses all 50,000 training images, scaled to $[0, 1]$. Final evaluation uses the standard test sets. We test latent dimensions $\{8, 32, 64\}$ for MNIST and $\{8, 32\}$ for CIFAR-10.

As in Song et al. (2020); Zhou et al. (2020), we train for 100,000 updates with batch-size 128. We use learning raetes 10^{-3} for MNIST and 10^{-4} for CIFAR-10. The objective the WAE uses combines squared reconstruction error with a latent regularization term, using a standard Gaussian prior. A plot of the convergence of the reconstruction error over the 100k steps is in Figure 20. The estimated latent score supplies the score at the end of the forward pass, and is then held fixed when differentiating the autoencoder objective, see Section E.6.2.

E.6.2 Minibatch electrostatic model fitting

Within each minibatch, we use the joint Lyapunov fit in Section B.3, with fixed K because there is no within-batch validation search. We construct a dictionary with 72 slab boundaries for $d = 8$ and 192 for $d \in \{32, 64\}$, constructed using ICA and linear Stein-residual directions. We combine it with 24 spherical centers selected by k-means and a shared width equal to half the median pairwise latent distance. We use regularization strength 0.01 for both families, with ordinary ℓ_2 penalties for slabs and feature-normalized ℓ_2 penalties for spheres, and hold the fitted score fixed when differentiating the autoencoder objective. More careful selection of any of these design choices could lead to better model results on the MNIST and CIFAR-10 datasets.

F Additional experimental results

F.1 Full two-dimensional baseline comparison

Table 4 gives the external baselines from which the "Best Baseline" in Table 2 chose from based on lowest test MSE for a given synthetic dataset.

Method	Banana	GMM	X-shape
Affine	1.317 ± 0.022 $K = 0$	1.845 ± 0.012 $K = 0$	14.890 ± 0.085 $K = 0$
SSGE	0.180 ± 0.102	0.178 ± 0.009	2.263 ± 0.173
SSM-VR	0.094 ± 0.010	$\mathbf{0.121 \pm 0.012}$	1.924 ± 0.181
DKEF	$\mathbf{0.074 \pm 0.011}$	0.138 ± 0.006	2.382 ± 0.186
Curl-free IMQ	0.462 ± 0.171	0.150 ± 0.015	$\mathbf{1.541 \pm 0.097}$
Gaussian KEF-CG	0.239 ± 0.075	0.158 ± 0.009	2.013 ± 0.362

Table 4: Two-dimensional baseline score errors on the same five runs as Table 2. Entries are mean test MSE $\pm$ one standard error. Bold identifies the lowest baseline mean for each distribution.

F.2 Dense dictionaries

Table 5 compares dense fits with their finite-grid test oracles on the same five runs as Table 2. Each dense fit uses all 1,600 spherical components, all 1,600 slab boundaries, or their 3,200-component

combination of the two. The linear field and all coefficients are fit jointly. All other methods are specificied in Section E.3.

Adaptive models retain only 18–145 components on average, compared with a much larger K in the full dictionaries. They achieve lower mean error on GMM for all three geometries and on X-shape for the slab and mixed models, while the full dictionaries perform better on Banana. Mixing geometries gives the lowest full-dictionary oracle error on each distribution, but validation does not always recover this advantage. Thus, a larger dictionary can improve approximation without improving the selected fit.

Method	Banana	GMM	X-shape
Full Spherical: selected	$\mathbf{0.079 \pm 0.009}$ $K = 1600$	0.148 ± 0.012 $K = 1600$	1.800 ± 0.092 $K = 1600$
Full Spherical: grid oracle	0.079 ± 0.009 $K = 1600$	0.140 ± 0.010 $K = 1600$	1.711 ± 0.070 $K = 1600$
Full Slabs: selected	0.194 ± 0.058 $K = 1595.8$	0.155 ± 0.009 $K = 1585.2$	$\mathbf{1.628 \pm 0.422}$ $K = 1599.8$
Full Slabs: grid oracle	0.137 ± 0.015 $K = 1595.8$	0.155 ± 0.009 $K = 1585.2$	1.253 ± 0.099 $K = 1599.8$
Full Mixed: selected	0.163 ± 0.058 $K = 3195.8$	0.134 ± 0.014 $K = 3185.2$	1.694 ± 0.093 $K = 3199.8$
Full Mixed: grid oracle	$\mathbf{0.073 \pm 0.007}$ $K = 3195.8$	$\mathbf{0.124 \pm 0.011}$ $K = 3185.2$	$\mathbf{1.185 \pm 0.064}$ $K = 3199.8$
Adaptive spherical: selected	0.102 ± 0.028 $K = 41.6$	0.142 ± 0.011 $K = 44.8$	2.049 ± 0.099 $K = 60.8$
Adaptive slabs: selected	0.292 ± 0.094 $K = 62.4$	0.132 ± 0.019 $K = 17.6$	1.377 ± 0.066 $K = 115.2$
Adaptive mixed geometries: selected	0.234 ± 0.086 $K = 130.2$	$\mathbf{0.088 \pm 0.012}$ $K = 21$	1.583 ± 0.161 $K = 145.2$

Table 5: Full-dictionary two-dimensional fits: validation-selected models and finite-grid test oracles. Entries are mean test MSE $\pm$ one standard error across five runs, with mean nonzero component count K. Bold identifies the lowest mean separately among selected fits and among oracles.

Table 6 shows the computational benefit of smaller dictionaries at $d = 30$: fitting the selected mixed dictionary is approximately 26–60 times faster than fitting its full counterpart. These results motivate adaptive fitting as a compact, competitive choice, although the timings exclude the cost of finding the dictionary.

Model	Banana	GMM	X-shape
Adaptive spheres	0.0784 ± 0.0085	0.0615 ± 0.0084	0.0246 ± 0.0050
Full spheres	5.2989 ± 0.1707	4.6295 ± 0.0588	5.1373 ± 0.3177
Adaptive slabs	0.1787 ± 0.0423	0.1293 ± 0.0464	0.5600 ± 0.1292
Full slabs	4.0190 ± 0.1216	3.9769 ± 0.1521	4.0360 ± 0.1205
Adaptive mixed	0.2466 ± 0.0362	0.2755 ± 0.0774	0.5828 ± 0.1274
Full mixed	14.7274 ± 0.7981	14.8737 ± 0.7319	15.1751 ± 0.4969

Table 6: CPU fitting time in seconds at $d = 30$ and $n = 1,600$ (mean $\pm$ SE, five seeds). Timings include computation and joint coefficient and liner base fitting, but exclude dictionary construction and model selection.

F.3 Synthetic dimension sweeps

We evaluate $d \in \{5, 10, 15, 30, 50, 75, 100\}$ with $1,600$ fitting, 400 validation, and $5,000$ test observations per run. All curves are total test score MSE (Eq. 39) averaged over five seeds with ± 1 SE. The distributional constructions below extend the two-dimensional distributions in Section E.1 to higher dimensions, with the goal of extending intrinsic complexity beyond 2D.

F.3.1 GMM HIGHER-DIMENSIONAL CONSTRUCTION

Motivated by the dimension-dependent Gaussian mixtures used by Sutherland et al. (2018); Zhou et al. (2020), we increase the number of components with d. Unlike their grid benchmark, our centers all lie at radius 5:

$$\begin{aligned} \mathcal{M}_d = &\left\{ \frac{5}{\sqrt{2}}(a\boldsymbol{e}_1 + b\boldsymbol{e}_2) : a, b \in \{-1, +1\} \right\} \\ &\cup \{\pm 5\boldsymbol{e}_j : j = 3, \dots, d\}, \\ p_{\mathrm{G}}^{(d)}(\boldsymbol{x}) = &\frac{1}{2d} \sum_{\boldsymbol{\mu} \in \mathcal{M}_d} \varphi_d(\boldsymbol{x}; \boldsymbol{\mu}, \mathbf{I}_d). \end{aligned} \tag{43}$$

Thus $d = 2$ recovers the four-component GMM, and each additional coordinate adds two components.

F.3.2 BANANA HIGHER-DIMENSIONAL CONSTRUCTION

Let $h(z) = (z^2 - 1)/2$ and $\boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. For even d, apply the two-dimensional Banana-like shape independently to consecutive coordinate pairs:

$$X_{2j-1} = Z_{2j-1}, \qquad X_{2j} = Z_{2j} + h(Z_{2j-1}), \quad j = 1, \dots, d/2. \tag{44}$$

For odd $d \geq 5$, we use the same Banana on coordinates $(1, 2)$, set

$$(X_3, X_4, X_5) = \big(Z_3,\ Z_4 + h(Z_3),\ Z_5 + h(Z_3)\big),$$

and pair the remaining coordinates as $(6, 7), (8, 9), \dots$ et cetera.

F.3.3 X-SHAPE HIGHER-DIMENSIONAL CONSTRUCTION

Draw one sign $S \in \{-1, +1\}$ with equal probability, then draw $\boldsymbol{X} \mid S = s \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}_s)$, where $\boldsymbol{\mu} = (1, 1, 0, \dots, 0)^\top$ and

$$\boldsymbol{\Sigma}_s = \operatorname{diag}(\underbrace{\mathbf{C}_s, \dots, \mathbf{C}_s}_{\lfloor d/2 \rfloor \text{ blocks}}), \qquad \mathbf{C}_s = \begin{pmatrix} 1 & 0.95s \\ 0.95s & 1 \end{pmatrix}. \tag{45}$$

For odd d, we append the scalar block $1 + s/2$. We construct the blocks as independent, conditional on that sign.

F.3.4 ADDITIONAL RESULTS

Figure 7 extends the main GMM comparison in Section 5.2 to the Banana and X-shape dimension sweeps specified above in Section F.3, and Table 7 records the selected spherical and slab component counts.

Dimension	Banana	GMM	X-shape
5	26.6/139.2 (5/5)	57.6/0.0 (0/5)	64.0/298.6 (5/5)
10	37.8/58.6 (5/5)	64.0/51.2 (4/5)	64.0/272.0 (5/5)
15	44.8/138.4 (4/5)	64.0/94.6 (5/5)	64.0/139.4 (5/5)
30	57.6/181.6 (5/5)	64.0/142.0 (5/5)	19.2/392.0 (5/5)
50	51.2/347.2 (5/5)	64.0/383.2 (5/5)	10.4/386.4 (5/5)
75	54.4/38.4 (3/5)	48.0/399.0 (5/5)	17.0/539.0 (5/5)
100	64.0/0.0 (0/5)	46.4/433.2 (5/5)	23.2/708.4 (5/5)

Table 7: Mean numbers of spherical components/slab boundaries in the selected mixed model. Parentheses give the number of runs containing both families, out of five. Counts exclude the affine field.

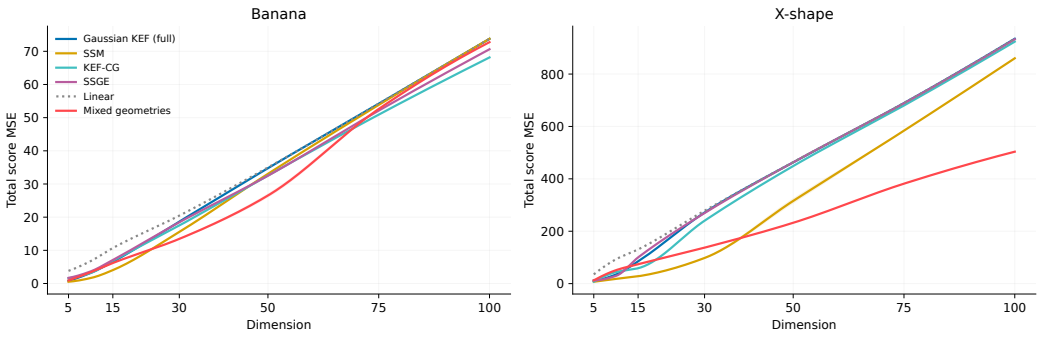


Figure 7: Mixed electrostatic models and baselines across dimension on Banana, GMM, and X-shape. Curves show mean total test score MSE over five runs; shading denotes one standard error. The GMM panel also appears in Figure 4.

F.4 Additional visualizations

F.4.1 Charge Geometry Placements

Figures 8 and 9 are the same spherical charge and slab charge placements from the learned and fitted models in Table 2. The supplement the X-shape only visualizations in the main document in Figure 3.

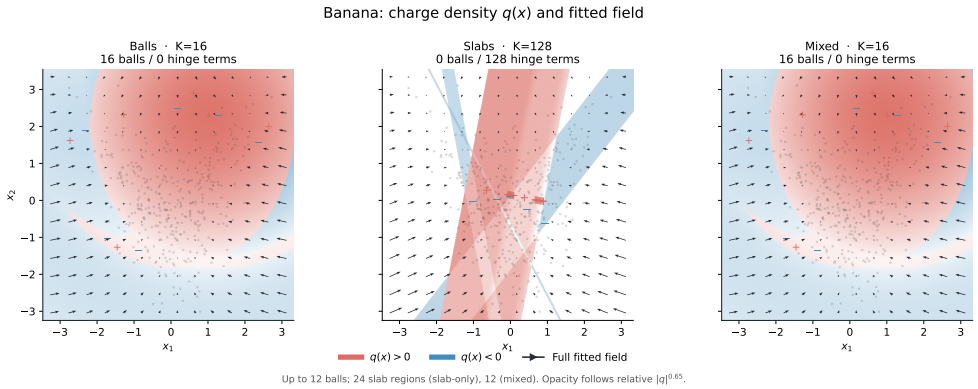


Figure 8: Selected charge components and complete score field on Banana. The mixed fit selects the spherical endpoint.

F.4.2 WAE Visuals: All Baselines

Visuals for the generated MNIST digits with the WAE at latent dimension $z = 8$, in Figure 10. We display generations by baseline models here; comparison with the best baseline, SSM, is in Section F.4.3.

F.4.3 WAE Visuals: SSM vs. Mixed Geometry

Additional visuals for MNIST and CIFAR-10 image generation using the Wasserstein Autoencoder. Visuals are from the proposed mixed geometry model, and compared with the best-performing baseline: SSM. See Figure 11–Figure 15.

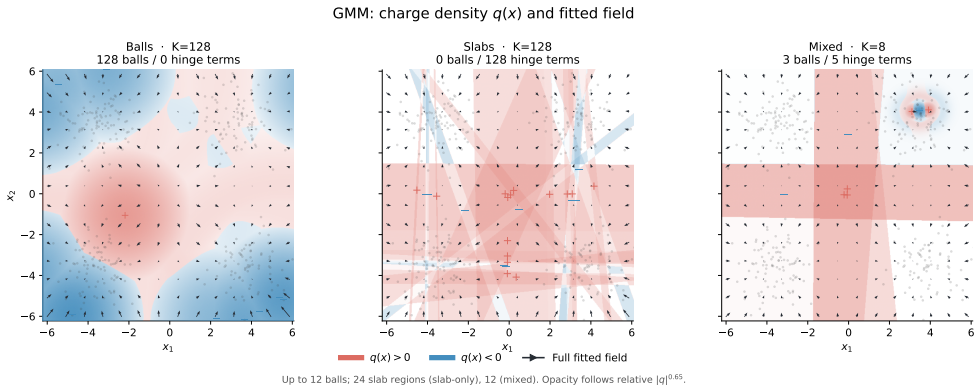
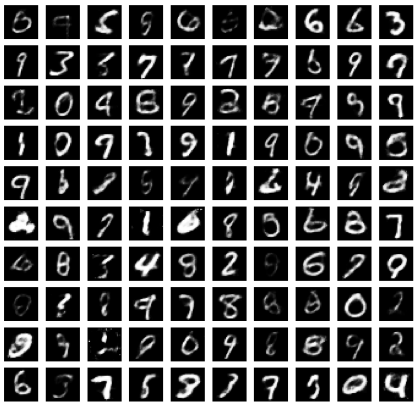
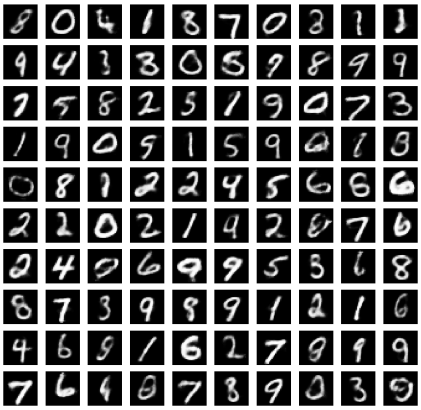


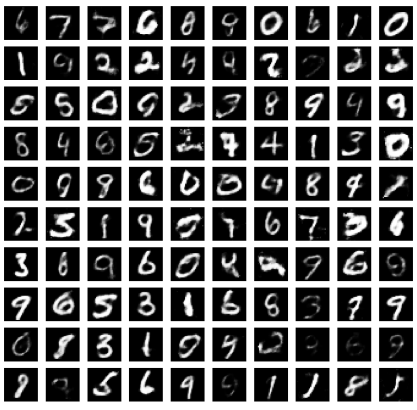
Figure 9: Selected charge components and complete score field on GMM. The mixed fit contains three spherical components and five slab boundaries; the latter are shown as their fitted halfspaces.



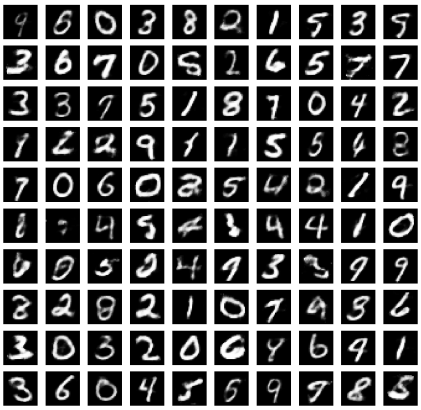
(a) Linear base score estimator



(b) SSM score estimator

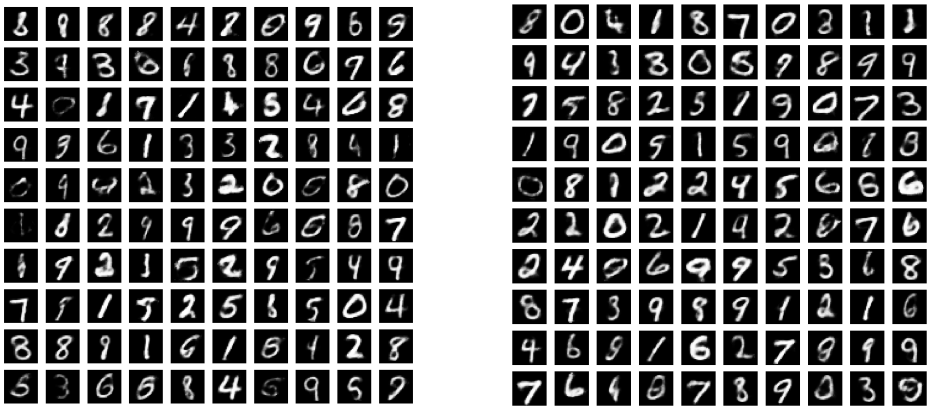


(c) SSGE score estimator



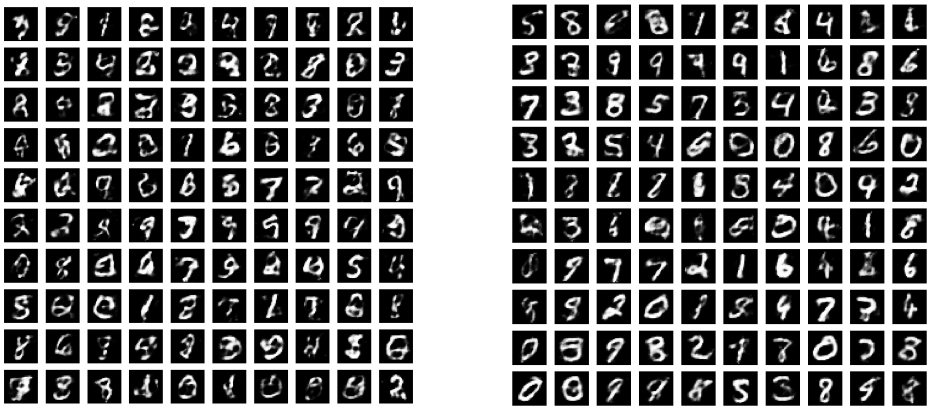
(d) KEF-CG score estimator

Figure 10: MNIST digit generation of with a WAE at latent dimension $z = 8$. Visual of baseline models. See Table 3 for NLL comparison.



(a) Mixed geometry score estimator (b) SSM score estimator

Figure 11: MNIST digit generation of with a WAE at latent dimension $z = 8$. Score estimation done by mixed geometry electrostatic model (left), and SSM (right). See Table 3 for NLL comparison.



(a) Mixed geometry score estimator (b) SSM score estimator

Figure 12: MNIST digit generation of with a WAE at latent dimension $z = 32$. Score estimation done by mixed geometry electrostatic model (left), and SSM (right). See Table 3 for NLL comparison.

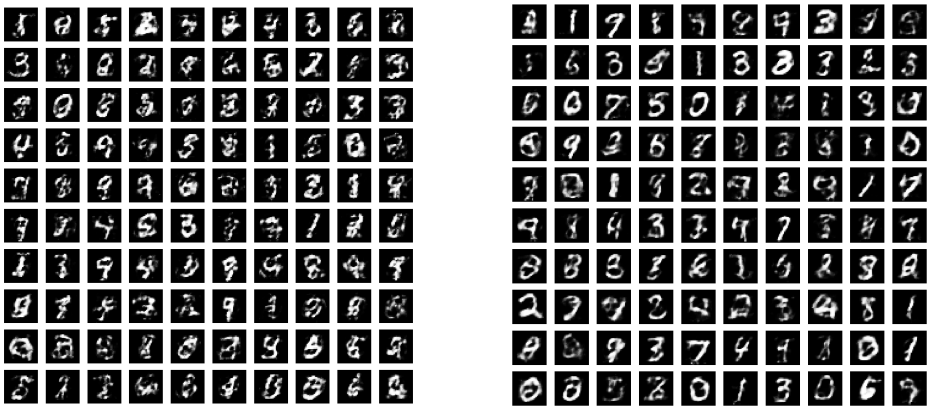
G Ablations and diagnostics

G.1 Selection along a regularization path

Selection of optimal hyperparameters remains a limitation. Using a small validation set can lead to variability in choice of regularization parameters $\{\boldsymbol{\lambda}\}$. Figure 16 demonstrates a successful and unsuccessful case with the same selection mechanism. At width 0.7, both selectors choose strength 0.1 although the plotted test minimum is at 0.01. At width 1.4, both choose 0.01. The plot illustrates a selection gap; test curves were never used to change either choice, just as a diagnostic at different seeds.

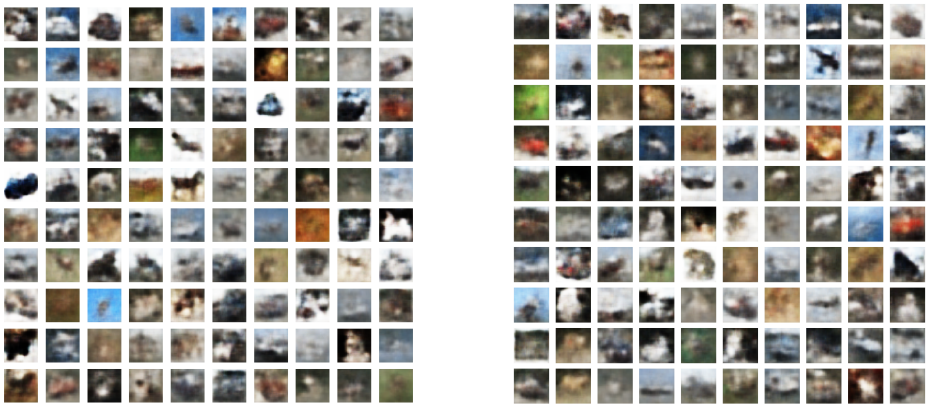
G.2 Regularization of spherical components

A design choice that can distinguish our spherical charge component model from the family of Gaussian KEF models is the choice of regularization. We explored different regularization methods, including the standard KEF RKHS regularization, and ℓ_2 regularization. Table 8 demonstrates results on the 2D distributions over five seeds between these two choices at a validation-selected regularization parameters. The exact equivalence between the two models only applies after this



(a) Mixed geometry score estimator (b) SSM score estimator

Figure 13: MNIST digit generation of with a WAE at latent dimension $z = 64$. Score estimation done by mixed geometry electrostatic model (left), and SSM (right). See Table 3 for NLL comparison.



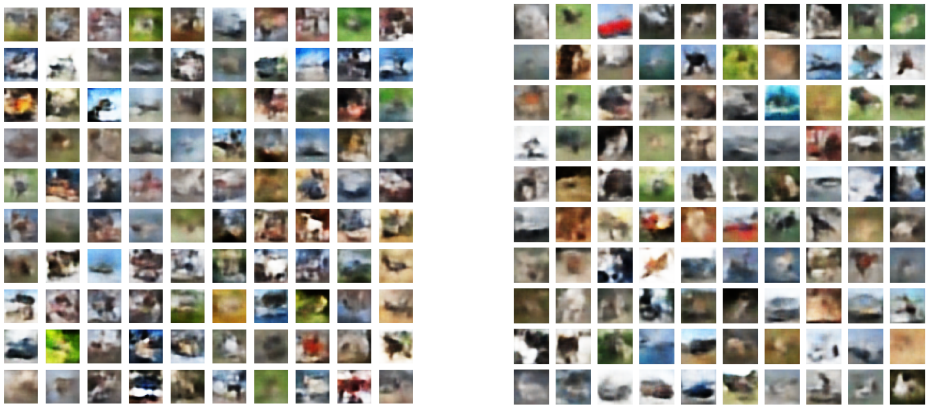
(a) Mixed geometry score estimator (b) SSM score estimator

Figure 14: CIFAR-10 image generation of with a WAE at latent dimension $z = 8$. Score estimation done by mixed geometry electrostatic model (left), and SSM (right). See Table 3 for NLL comparison.

regularization (and the bases) are equivalent. We chose ℓ_2 regularization for simplicity in applying it to both the uniformly charged slabs and spherically charged features. This will account for differences between the Gaussian KEF method and the spherically charged models. Additional differences apply when the bases are different, which occurs in the adaptive sphere model, but not the "full-dictionary" models.

Method	Banana	GMM	X-shape
Feature-normalized ℓ_2	$\mathbf{0.084 \pm 0.013}$	0.170 ± 0.016	$\mathbf{2.815 \pm 1.101}$
Gaussian RKHS	0.102 ± 0.008	$\mathbf{0.147 \pm 0.019}$	3.406 ± 1.758

Table 8: Dense spherical regularization comparison on two development runs. Plain validation loss selects width and strength separately for each penalty. Entries are mean test MSE $\pm$ one standard error.



(a) Mixed geometry score estimator (b) SSM score estimator

Figure 15: CIFAR-10 image generation of with a WAE at latent dimension $z = 32$. Score estimation done by mixed geometry electrostatic model (left), and SSM (right). See Table 3 for NLL comparison.

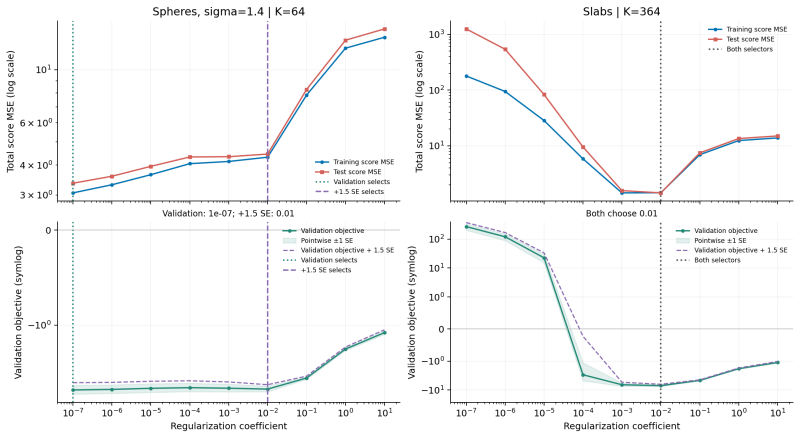


Figure 16: Regularization-path diagnostic on X-shape, for spherical (left) and slab (right) components. Top is Test MSE on a log scale, bottom is validation objective. Test score errors are computed after freezing the validation choices. Vertical lines mark plain validation selection and validation plus 1.5 standard errors, defined in eq. 42; overlapping lines indicate agreement, as in the top right.

G.3 Slab geometry construction and coefficient optimization

We consider different ways to optimize the slab geometry (directions $\{\boldsymbol{u}\}, \{\boldsymbol{b}\}$). Although the method used for the adaptive slab is the ICA/Stein directions described in Section E.3, below we compare alternate approaches for choosing the weights, as well as other similar-geometries. Each row in Table 9 is:

- **Same geometry + ReLU SGD**: This uses the same ICA/Stein fit geometry, but optimizes the learned coefficients $\boldsymbol{\theta}$ with a ReLU SGD.
- **ICA/Stein + quadratic fit**: Model detailed as adaptive slab in Section E.3.
- **Tied ReLU-SGD**: This model ties the input and output vectors of a ReLU MLP (see Section D.3 for construction and similarity to slab model), and is optimized with SGD. Its geometry therefore differs from the ICA/Stein choice of directions and breakpoints, but its construction is still made from slabs.
- **Tied Softplus**: This uses the softplus activation function instead of ReLU, and therefore is not technically equivalent to the slab model, even though very similar in terms of its structure.

- **Softplus + Slab**: reuses learned directions and offsets, and then refits the hard-ReLU coefficients.

These rows in Table 9 distinguish geometry quality from the coefficient solver; they are not pooled with the main dimension results.

Method	Test MSE (SE)	Conditional fit, s
Same geometry + ReLU SGD	8.873 ± 1.269	35.609
ICA/Stein + quadratic fit	8.832 ± 1.290	**0.033**
Tied ReLU-SGD	14.504 ± 0.199	18.578
Tied Softplus	6.653 ± 0.224	22.024
Softplus + Slab	$\mathbf{6.417 \pm 0.288}$	0.038

Table 9: Separate exploration on an X-Shape distribution, generated with 2D complexity, then rotated and raised with Gaussian noise to higher dimensions. This is for comparison of coefficient optimization methods; it is not meant to be compared to other sections of this paper. **Specifications:** X-Shape $D = 10$ with complexity rank $r = 2$: $1,600/400/2,000$ observations and seeds 11,22,33,44,55. MSE is mean $\pm$ SE. Conditional fit time is the archived mean in seconds on A100-SXM4-80GB, with geometry fixed where applicable; learning-rate search is excluded. Bold marks the best value in each numerical column. This protocol is separate from the full-rank dimension sweep.

G.4 Kernel models across dimension

We established the similarity between Gaussian KEFs and our spherically charged component model in Section D.2. The difference in the implementation of the Gaussian KEF (full) and the full-dictionary spherical model is the use of RKHS vs ℓ_2 regularization, respectively, and the fitting of the linear base. These curves deliberately compare the implemented estimators: scalar KEF fixes its fitted Gaussian base, whereas adaptive spherically symmetric charge elements jointly fit a symmetric linear base field. KEF-CG uses a derivative-representer construction and a different kernel (Section D.2.1). Thus differences here cannot be attributed solely to coefficient coordinates. For kernel methods, in higher dimensions, the selected spherical charge corrections often remain close to the linear base field; this trend is visible in the non-kernel methods as well in Figure 4.

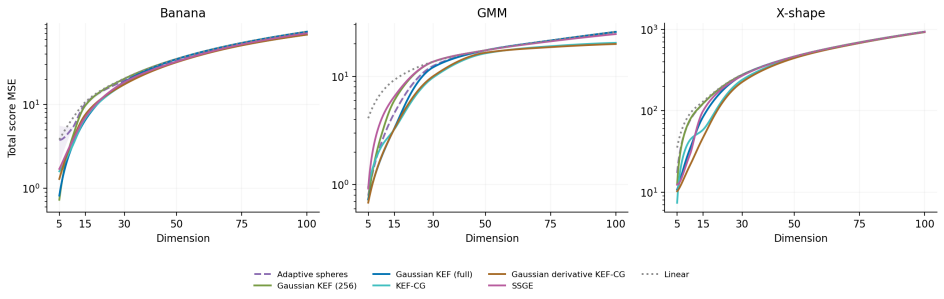


Figure 17: Matched full-rank kernel comparison: mean total score MSE $\pm$ one SE over five seeds. Gaussian KEF uses either all $1,600$training centers or 256 centers to provide a closer comparison for the adaptive spheres; adaptive spheres use at most $K = 64$ from 128 proposals. Linear base is the reference without geometric corrections ($K = 0$).

G.5 Fixed-configuration fitting time

See Figure 19 for the training time plot over a single seed across dimensions for each model in Figure 4. Time is in seconds on CPU on a log–log scale. Mixed geometries recomputes training moments and jointly fits coefficients and the symmetric linear base field on its archived selected dictionary. SSM trains to a pre-selected step to keep validation removed from the training times.

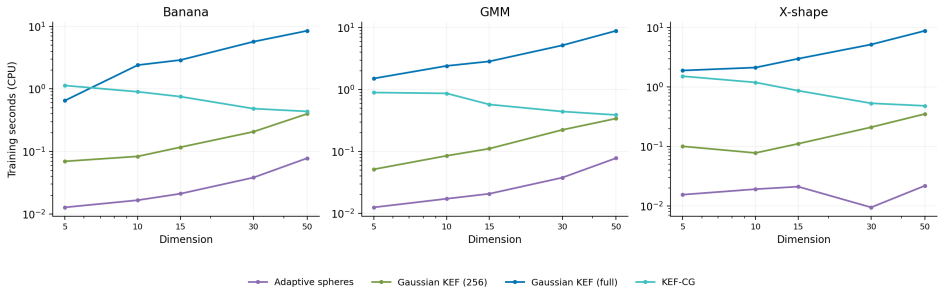


Figure 18: Model fitting time on the same CPU on one seed, in log–log axes. Each fit recomputes training and solves at the selected settings. In this timing comparison, spherically charged component parameters are fixed; construction, adaptive search, validation, selection, and test evaluation are excluded. Equivalent parameters in the kernel models are also fixed for a fair comparison of fitting times. These are not A100 or end-to-end tuning times.

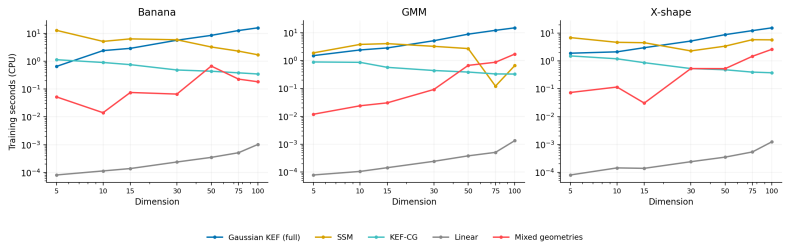


Figure 19: Fixed-configuration training for all five main dimension-plot models, on the same CPU on one seed. Plotted with a log–log axes across a dimension sweep 5–100. For all models, construction, adaptive search, validation and hyperparameter selection are excluded.These conditional fitting times do not measure complete adaptive training pipelines.

G.6 WAE Convergence

See Figure 20 for the convergence plots of the training reconstruction error over the 100k training update steps.

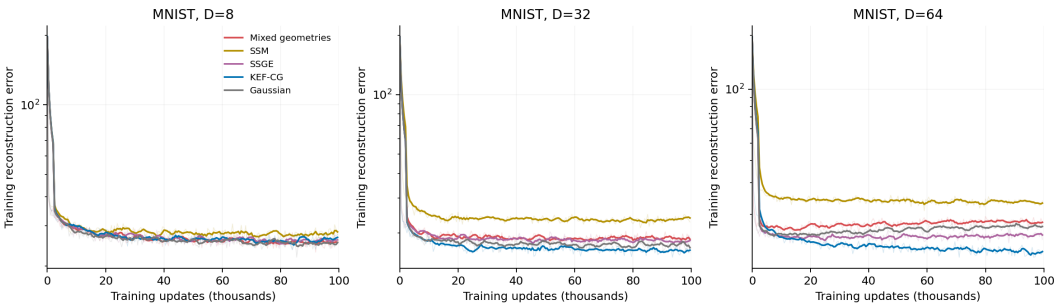


Figure 20: WAE convergence of models in Table 3 on the 100k steps.