Learnable Location–Scale Cauchy–KL Kernels: Theory and a Controlled Regression Study

Heyang Gong

Preprint source prepared July 22, 2026

Evidence status. The mathematical results in this manuscript are independent of the numerical experiments. The current `corrected-v3` implementation enforces the intended perception $\to$ μ $\to$ method-specific predictor contract. A complete 315-row candidate and separate five-seed input- and label-robustness artifacts now pass their structural gates. Within each learned-kernel method, query-local top-k KRR lowers mean RMSE relative to global KRR in all 27 corrected cells. They show no general Deep Cauchy advantage over matched Deep RBF on clean prediction, input contamination, or label noise. Two narrower local mechanisms are supported: selection rejects grossly corrupted stored inputs, while local solves limit the propagation of gross label swaps but not diffuse Gaussian target noise. All numerical intervals remain exploratory. The version boundary is summarized in Appendix A.8 and specified in Appendix I. A separate 70-row `classification-v1-exploratory` screen on Breast Cancer and Digits is reported only in the appendix; it is not part of the `corrected-v3` regression matrix and does not change the regression claims.

Abstract

We study a learnable location–scale kernel obtained by mapping each input to a product of univariate Cauchy distributions with learned location $\mu(x)$ and positive scale $\gamma(x)$, then exponentiating their exact Kullback–Leibler divergence. We prove that the resulting Cauchy–KL construction is positive semidefinite for arbitrary learned parameter maps, give a sufficient condition for strict positive definiteness, identify its fixed-scale reduction, derive its local pullback geometry, and differentiate the associated kernel ridge regression (KRR) solution. For controlled empirical evaluation, all learned methods use the same perception-to-μ neural structure as closely as possible: Direct MLP continues with a scalar output, Deep RBF continues with RBF KRR, and Deep Cauchy adds a parallel γ head before Cauchy–KL KRR. A fixed-scale Deep Cauchy ablation isolates the contribution of input-dependent scale. The complete `corrected-v3` candidate shows no general clean-RMSE advantage for Deep Cauchy over Deep RBF or the fixed-scale ablation. Its strongest positive empirical result is instead architectural: for the same learned representation, kernel, and targets, query-local top-k KRR lowers mean RMSE relative to global KRR in every corrected dataset–sample-size–kernel cell. Paired 10% and nested 20% dirty-development screens show no stable Cauchy-specific robustness. A frozen-model reference-bank isolation further shows that local selection removes grossly corrupted stored references for RBF and Cauchy alike. In a separate 20% label-noise screen, noisy labels remain inside local neighborhoods, but local KRR degrades less than global KRR under label swaps in all nine learned-kernel cells; Gaussian target noise shows no such pattern. The supported mechanism is therefore learned local reference selection and influence localization, not universal Cauchy superiority.

1 Introduction

Kernel similarity determines which observations a regression method treats as nearby. Standard stationary kernels use one global length scale, while deep kernel learning changes geometry through a learned representation [19]. Neither choice by itself provides a mathematically safe way to learn an input-dependent local scale: inserting arbitrary local scales into a familiar radial formula need not preserve positive semidefiniteness.

The Cauchy location–scale family provides a special route. Univariate Cauchy KL divergence is symmetric and its square root admits a Hilbert-space embedding [13]. Exponentiation therefore produces a valid kernel by Schoenberg's theorem [16]. We extend this construction to product distributions and arbitrary learned maps $x \mapsto (\mu(x), \gamma(x))$, while separating positive semidefiniteness from strict positive definiteness and deriving the induced local geometry.

The empirical question is whether this additional scale flexibility is useful, not merely whether the kernel is valid. Every learned method independently instantiates the same perception architecture $h_\theta : \mathbb{R}^d \to \mathbb{R}^r$ followed by a linear μ head. The methods then diverge only at their intended prediction mechanism: a scalar neural output, RBF KRR, fixed-scale Cauchy–KL KRR, or Cauchy–KL KRR with a parallel input-dependent γ head. This design separates three questions:

1. Is the learned product Cauchy–KL construction a valid kernel, and what local geometry does it induce?
2. Does an input-dependent $\gamma(x)$ add value beyond a learned global Cauchy scale?
3. Does Cauchy location–scale geometry add value beyond RBF geometry on the same perception-to-μ neural structure?

Direct prediction, fixed input-space kernels, and local versus global KRR are contextual controls rather than separate paper identities. The positive scale $\gamma(x)$ is a similarity parameter, not automatically calibrated uncertainty, and clean predictive accuracy does not answer robustness under contaminated development data; these interpretation boundaries are collected in Appendix A.

The complete `corrected-v3` candidate and separate robustness artifacts now pass their structural gates. We report their heterogeneous five-seed results as exploratory evidence without promoting them to superiority or equivalence. The compact status rule is in Appendix A.8; the executable acceptance contract is in Appendix I.

Contributions.

- We formulate a learnable product Cauchy–KL kernel and prove its positive-semidefinite validity for arbitrary learned location and scale maps, with a sufficient distinctness condition for strict positive definiteness on finite samples.
- We characterize the construction through its fixed-scale reduction, local pullback geometry, and derivatives needed for end-to-end KRR.
- We define a controlled comparison in which neural methods share the perception-to-μ structure as closely as possible, using matched RBF, fixed-scale Deep Cauchy, and Direct-MLP controls to separate kernel geometry, input-dependent scale, and prediction mechanism.
- We show that, conditional on the learned kernel geometry, query-local top-k KRR consistently improves over its global-KRR counterpart and can exclude grossly corrupted stored inputs from the local solve or limit the propagation of gross label swaps. These mechanisms do not produce a general Cauchy-specific advantage.

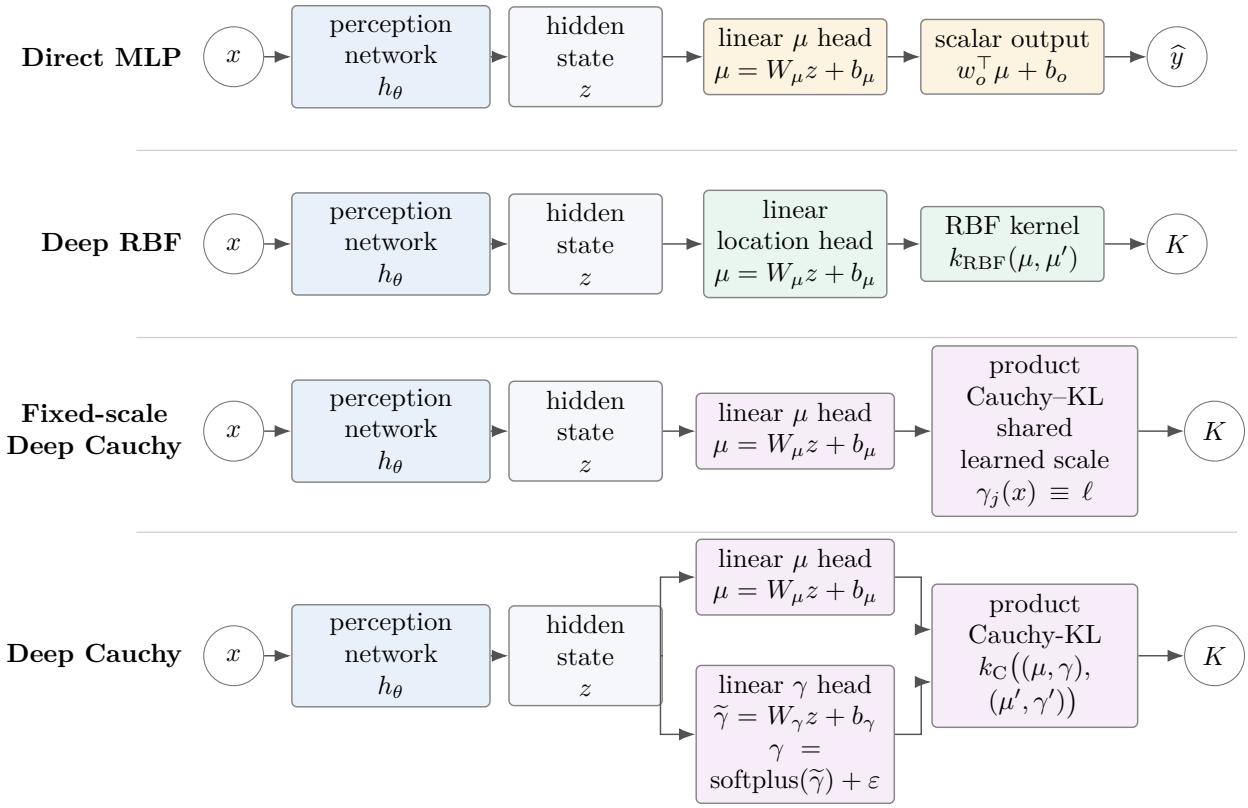

Figure 1: Architectures of the four learned predictors. Every method retains the same perception-to-μ neural structure as closely as possible, then continues with its method-specific predictor. Fixed-scale Deep Cauchy learns one input-invariant scale ℓ; Deep Cauchy adds the parallel positive $\gamma(x)$ head being tested. Architecture, dimensions, and per-seed initial values along the perception-to-μ path are matched, but each method independently trains its own weights.

2 Model

2.1 Learned representation and kernel parameters

The consolidated notation table in Appendix B records the domains, shapes, and roles of the symbols used below. In particular, it distinguishes the benchmark sample size n, reference-bank size N, training solve size B, and inference neighborhood size k.

Given data $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, a backbone produces $z_i = h_\theta(x_i) \in \mathbb{R}^r$. The RBF model uses a linear location head

$$\mu_\theta(x) = W_\mu h_\theta(x) + b_\mu \in \mathbb{R}^p. \tag{1}$$

The Cauchy model adds a positive scale head

$$\gamma_\theta(x) = \mathrm{softplus}(W_\gamma h_\theta(x) + b_\gamma) + \varepsilon \in \mathbb{R}^p_{>0}, \qquad \varepsilon > 0, \tag{2}$$

and associates x with the component-independent distribution

$$P_x = \bigotimes_{j=1}^{p} \mathrm{Cauchy}(\mu_{\theta,j}(x), \gamma_{\theta,j}(x)) \,. \tag{3}$$

Architecture-matched neural paths. The method index is suppressed above for readability. In the empirical comparison, each neural method a owns an independent parameter instance and computes

$$z_a(x) = h_{\theta_a}(x), \qquad \mu_a(x) = W_{\mu,a} z_a(x) + b_{\mu,a}. \tag{4}$$

The backbone and linear μ head have the same architecture, dimensions, and per-seed initial values across methods, but their learned weights are not tied. The method-specific continuations are

$$\text{Direct MLP:} \quad \widehat{y}_a(x) = w_{o,a}^\top \mu_a(x) + b_{o,a}, \tag{5}$$

$$\text{Deep RBF:} \quad \mu_a \longrightarrow k_{\mathrm{RBF}} \longrightarrow \mathrm{KRR}, \tag{6}$$

$$\text{Fixed-scale Deep Cauchy:} \quad (\mu_a, \ell_a) \longrightarrow k_{\mathrm{C}} \longrightarrow \mathrm{KRR}, \tag{7}$$

$$\text{Deep Cauchy:} \quad (\mu_a, \gamma_a) \longrightarrow k_{\mathrm{C}} \longrightarrow \mathrm{KRR}, \tag{8}$$

where $\gamma_a(x) = \mathrm{softplus}(W_{\gamma,a} z_a(x) + b_{\gamma,a}) + \varepsilon$ is a parallel Cauchy-only branch. Thus Direct MLP does not map the backbone output straight to a scalar, and Deep Cauchy is not total-parameter matched to Deep RBF because it intentionally adds γ. The two linear maps in the Direct path could be collapsed algebraically. We retain the explicit μ head so that Direct MLP shares the same perception-to-μ neural structure with the kernel methods as closely as possible; it is not included to increase the Direct model's function-class expressivity. The complete interpretation and reproducibility contracts are in Appendices A.2 and I.2.

The implemented backbone is an MLP; none of the kernel-validity results require the backbone to be invertible.

2.2 RBF and Cauchy-KL kernels

The RBF comparison is

$$k_{\mathrm{RBF}}(x, x') = \exp\left(-\frac{\|\mu_\theta(x) - \mu_\theta(x')\|_2^2}{2\ell^2} \right), \qquad \ell > 0. \tag{9}$$

For two univariate Cauchy laws, define

$$d_{\mathrm{C}}((\mu, \gamma), (\mu', \gamma')) = \log \frac{(\mu - \mu')^2 + (\gamma + \gamma')^2}{4\gamma\gamma'}. \tag{10}$$

The product Cauchy-KL kernel is

$$D_{\mathrm{C}}(x, x') = \sum_{j=1}^{p} d_{\mathrm{C}}((\mu_j(x), \gamma_j(x)), (\mu_j(x'), \gamma_j(x'))), \tag{11}$$

$$\begin{aligned} k_{\mathrm{C}}(x, x') &= \exp(-\beta D_{\mathrm{C}}(x, x')) \\ &= \prod_{j=1}^{p} \left[\frac{4\gamma_j(x)\gamma_j(x')}{(\mu_j(x) - \mu_j(x'))^2 + (\gamma_j(x) + \gamma_j(x'))^2} \right]^{\beta}, \qquad \beta > 0. \end{aligned} \tag{12}$$

A scale ablation replaces every $\gamma_j(x)$ by one learned global scalar ℓ. It therefore retains the Cauchy kernel family while removing input-dependent and dimension-dependent scale information.

Theorem 1 (Kernel validity)**.** *For every $\beta > 0$ and every map $x \mapsto (\mu_\theta(x), \gamma_\theta(x)) \in \mathbb{R}^p \times \mathbb{R}^p_{>0}$, the function in Eq. (12) is a positive-semidefinite kernel. For a finite set $x_1, \ldots, x_m$, its Gram matrix is strictly positive definite whenever the joint parameter tuples $(\mu_\theta(x_i), \gamma_\theta(x_i))$ are pairwise distinct.*

The proof is in Appendix E. The distinction matters: an arbitrary neural map can collapse two inputs to the same distribution, in which case no kernel on those distributions can yield a strictly positive-definite Gram matrix for the duplicated pair.

2.3 Global and local KRR

For a positive-semidefinite Gram matrix K and $\lambda > 0$, global KRR uses

$$\alpha = (K + n\lambda I_n)^{-1} y, \qquad \widehat{f}(x_*) = k_*^\top \alpha. \tag{13}$$

For local inference, let $\mathcal{N}_k(x_*)$ be the indices of the k largest similarities $k_\theta(x_*, x_i)$. With $K_\mathcal{N}$ denoting the Gram matrix on that neighborhood,

$$\alpha_{x_*} = (K_\mathcal{N} + k\lambda I_k)^{-1} y_\mathcal{N}, \qquad \widehat{f}_{\text{local}}(x_*) = k(x_*, X_\mathcal{N})^\top \alpha_{x_*}. \tag{14}$$

The ridge term makes both systems uniquely solvable even when the kernel is only semidefinite. Local versus global inference is a secondary evaluation factor rather than the paper's defining method. The selected reference set is inspectable, although the KRR weights need not be nonnegative and should not be interpreted causally. A pipeline diagram and the distinction among N, B, and k appear in Appendix A.6.

2.4 End-to-end training

The deep-kernel models backpropagate mean squared error through the linear solve in Eq. (13), following the general differentiable-solver principle used in meta-learning [1, 9]. The `corrected-v3` reference protocol uses full-batch deep-kernel training: it constructs the training Gram matrix, solves for α, and evaluates MSE on that training partition without top-k selection. The Direct MLP instead uses shuffled mini-batch scalar MSE. Validation recomputes the kernel and ridge solution after every optimizer update so that the neural and KRR states refer to one parameter version. Only after training, with neural parameters frozen, does local inference select top-k references. The derivative identity, stage-by-stage computation, sample-count conventions, and discrete-selection boundary are given in Appendices A.6 to A.7 and G.

3 Mathematical properties

3.1 Exact divergence and product additivity

For $P = \text{Cauchy}(\mu, \gamma)$ and $Q = \text{Cauchy}(\mu', \gamma')$, direct integration gives

$$D_{\text{KL}}(P\|Q) = d_{\text{C}}((\mu, \gamma), (\mu', \gamma')). \tag{15}$$

Unlike a generic KL divergence, this expression is symmetric on the univariate Cauchy location-scale family. For product distributions, KL additivity gives

$$D_{\text{KL}}\left(\bigotimes_j P_j \middle\| \bigotimes_j Q_j\right) = \sum_j D_{\text{KL}}(P_j\|Q_j). \tag{16}$$

The integration and additivity arguments are supplied in Appendix D.

3.2 Fixed-scale reduction

Corollary 2 (Fixed-scale kernel)**.** *When $p = 1$, $\mu(x) = x$, and $\gamma(x) \equiv \ell > 0$,*

$$k_{\text{C}}(x, x') = \left(1 + \frac{(x - x')^2}{4\ell^2}\right)^{-\beta}. \tag{17}$$

Thus the learned-scale construction contains a generalized Cauchy, or rational-quadratic-type, stationary kernel as a special case.

3.3 Local pullback geometry

The Cauchy construction also determines what "local scale" means. Consider a smooth path $t \mapsto (\mu_j(t), \gamma_j(t))$. Then

$$D_{\mathrm{C}}(t, t+\delta) = \frac{\delta^2}{4} \sum_{j=1}^{p} \frac{\dot{\mu}_j(t)^2 + \dot{\gamma}_j(t)^2}{\gamma_j(t)^2} + O(\delta^3). \tag{18}$$

Consequently, $-\log k_{\mathrm{C}} = \beta D_{\mathrm{C}}$ locally penalizes movement in location and scale, normalized by the squared Cauchy scale. For the original one-dimensional parameterization $\mu(x) = x$ and $\gamma(x) = g(x)$, the coefficient reduces to $\beta(1+g'(x)^2)/(4g(x)^2)$. A complete Taylor expansion appears in Appendix F.

4 Related work

Distribution kernels compare objects through probability laws. Probability product kernels construct direct Hilbert-space inner products [7]; our construction instead uses the special negative-definite geometry of univariate Cauchy KL [13]. Input-dependent covariance kernels are also well established in nonstationary Gaussian-process modeling [14]. We therefore do not interpret input-dependent scale by itself as a novelty claim. The contribution is the valid learned product Cauchy–KL construction, its geometry, and a controlled test of whether its input-dependent scale adds predictive value.

Deep kernel learning combines neural representations with Gaussian-process or kernel inference [15, 19]. Differentiable closed-form linear and quadratic solvers provide a general route for training representations through downstream optimization [1, 9]. Our use of a KRR solve is an instance of that route rather than a new differentiation principle.

Local prediction predates modern deep learning. Classical examples include Nadaraya–Watson regression [11], locally weighted and local polynomial regression [3, 5], and local Gaussian processes [12]. Learned metrics and representations have also been combined with nearest-neighbor prediction [6, 8, 17, 18]. Sparse Kernels expose fixed or learned representations within a differentiable lazy kernel layer [10]. We do not claim to introduce the first local KRR layer; local inference is one evaluation setting for the learned Cauchy similarity.

5 Empirical questions and evaluation design

5.1 Corrected-v3 matrix

The `corrected-v3` matrix uses California Housing, Wine Quality, and Abalone; sample sizes $n \in \{500, 1000, 2000\}$; and five seeds. Feature preprocessing is fitted on the training partition only. The seven method identifiers are listed in Table 1. Kernel methods are evaluated with global and local inference where applicable. The resulting serialized reference contains $7 \times 3 \times 3 \times 5 = 315$ unique method rows.

Table 1: Methods in the `corrected-v3` matrix. All neural methods match the perception-to-μ path in architecture, dimensions, and per-seed initialization; each method is trained independently.

Identifier	Description
`gradient_boosting`	sklearn gradient-boosted regression trees
`fixed_rbf`	input-space RBF KRR
`fixed_cauchy`	input-space product Cauchy KRR
`direct_mlp`	backbone $\to$ linear $\mu \to$ scalar output
`deep_rbf`	backbone $\to$ linear $\mu \to$ RBF KRR
`deep_cauchy`	backbone $\to$ linear (μ, γ) heads $\to$ Cauchy KRR
`deep_cauchy_fixed`	backbone $\to$ linear $\mu \to$ fixed-scale Deep Cauchy KRR

5.2 Primary comparisons

All comparisons are paired by dataset, sample size, and seed. They are ordered by their relevance to the paper's central question:

1. **Input-dependent scale:** `deep_cauchy` versus `deep_cauchy_fixed` isolates whether $\gamma(x)$ adds value beyond one learned global Cauchy scale.

2. **Kernel geometry:** `deep_cauchy` versus `deep_rbf` compares Cauchy location–scale geometry with RBF geometry on the matched perception-to-μ path.

3. **Prediction mechanism:** the deep kernels versus `direct_mlp` compare KRR with a scalar neural continuation after the same explicit μ structure.

4. **Representation and inference controls:** deep versus fixed kernels and local versus global inference contextualize the primary contrasts without redefining the paper's identity.

The analyzer rejects duplicate, incomplete, or wrong-protocol matrices before forming aggregates. The exact architecture, preprocessing, and matrix acceptance rules are in Appendices I and I.2.

5.3 Current evidence boundary

A complete 315-row `corrected-v3` candidate passes its structural and provenance gates. Deep Cauchy has no general clean-RMSE advantage over matched Deep RBF: eight of nine dataset/size cells are indistinguishable in the five-seed screen, and the remaining California cell favors RBF. Input-dependent scale likewise has no stable advantage over the shared-scale Cauchy ablation. Direct MLP and tuned sklearn Gradient Boosting are competitive controls; their relative ranking is dataset-dependent. Fixed-kernel superiority claims remain gated because many validation grids select a boundary.

Separately, a 70-row exploratory classification screen covers Breast Cancer and Digits with five paired seeds and seven methods. It is reported in Appendix C.7, uses a classification-specific protocol and acceptance gate, and is not pooled with any regression comparison.

5.4 Dirty-development robustness and mechanism isolation

The robustness protocol corrupts raw training and validation inputs while keeping test inputs and all targets clean. The 10% screen and 20% severity follow-up use the same five seeds, methods, selected feature, $N = B = 1000$, and local $k = 100$. The 20% masks are nested: they retain every

frozen 10% corrupted row and add another 10 percentage points. Table 2 reports clean-test RMSE degradation at the higher severity.

Table 2: Dirty-development clean-test RMSE degradation at $\rho = 0.20$. Negative values are improvements relative to the paired clean model; negative Cauchy–RBF contrasts favor Cauchy. GBR is sklearn Gradient Boosting, not native XGBoost.

Dataset	corruption	Deep RBF	Deep Cauchy	Direct MLP	GBR	C–R
California	mixed units	+0.0629	+0.0653	+0.0478	+0.0148	+0.0025
California	centered sign	+0.0234	+0.0223	+0.0137	+0.0037	−0.0011
Wine	mixed units	+0.0152	+0.0142	+0.0100	+0.0113	−0.0011
Wine	centered sign	+0.0045	+0.0010	−0.0005	+0.0040	−0.0034
Abalone	mixed units	−0.0270	−0.0207	+0.0384	+0.0123	+0.0063
Abalone	centered sign	−0.0189	−0.0250	+0.0206	+0.0231	−0.0060

The direction of Cauchy minus RBF changes across datasets and corruptions at both severities. Five of six 20% screening intervals include zero; the small Wine centered-sign contrast is the sole exception. California worsens as severity rises, Wine is nearly flat, and Abalone deep local KRR remains resilient while Direct MLP and Gradient Boosting degrade. These heterogeneous five-seed results support neither a general Cauchy-specific robustness claim nor universal monotone degradation.

The separate reference-bank-only experiment freezes a clean-trained model, clean validation choice, and clean scaler, then corrupts only 10% of stored reference inputs. Under gross mixed-unit errors, nearly every designated reference leaves top-k for Deep RBF, Deep Cauchy, and shared-scale Deep Cauchy; neighborhood overlap remains about 0.815–0.821 and RMSE changes by only a few thousandths. This isolates query-local reference rejection as the supported mechanism. It does not attribute that robustness to Cauchy tails or learned scale.

5.5 Dirty-label robustness

A separate paired screen corrupts 20% of training and validation targets while keeping every input and the test split clean. The same row masks are shared across methods. Gaussian noise adds one clean-training target standard deviation; label swap permutes the designated targets without fixed points. Table 3 reports local-inference clean-test degradation.

Table 3: Clean-test RMSE degradation under 20% training/validation target noise. Lower is better; negative Cauchy–RBF contrasts favor Cauchy.

Dataset	noise	Deep RBF	Deep Cauchy	Fixed Cauchy	Direct MLP	GBR	C–R
California	Gaussian	+0.0256	+0.0181	+0.0307	+0.0239	+0.0405	−0.0076
California	label swap	+0.0811	+0.0801	+0.0755	+0.0806	+0.1040	−0.0010
Wine	Gaussian	+0.0105	+0.0076	+0.0097	−0.0033	+0.0127	−0.0029
Wine	label swap	+0.0200	+0.0206	+0.0218	+0.0075	+0.0171	+0.0006
Abalone	Gaussian	+0.0119	+0.0014	+0.0257	+0.0140	+0.0172	−0.0105
Abalone	label swap	+0.0977	+0.1097	+0.1094	+0.1332	+0.1679	+0.0120

Because target corruption does not make an input geometrically abnormal, approximately 19.2–20.2% of selected references have designated noisy targets and every test query is exposed to at least one. Local selection does not reject these rows. Under label swap, however, local KRR has

lower mean degradation than global KRR in all nine dataset–kernel cells; the paired interval excludes zero for all three kernels on California and Abalone (six of nine cells). Under Gaussian target noise, local degradation is larger in mean in eight of nine cells, but only California Deep Cauchy has an interval that excludes zero in the global-favoring direction. This is consistent with local inference containing a gross input–target mismatch, while a global solve can average diffuse zero-mean noise over more references. All six Cauchy-minus-RBF intervals include zero, so the screen gives no Cauchy-specific label-robustness advantage. The frozen-kernel direct-influence calculation is given in Appendix C.6.

6 Discussion and limitations

What the mathematics establishes. The product Cauchy–KL kernel is valid for arbitrary learned parameter maps, its fixed-scale case recovers a familiar stationary Cauchy form, and its local metric states exactly how changes in location and scale alter similarity. Ridge regularization makes each KRR solve well posed. These are deterministic statements independent of a dataset or optimizer. The regression and classification prediction rules that can be placed on top of the kernel are derived separately in Appendix C. The independent classification screen in Appendix C.7 establishes only that these predictors are executable on one binary and one ten-class dataset; it is not evidence of a general Cauchy classification advantage.

What query-local KRR contributes. The strongest positive empirical result is conditional but consistent. Holding the fitted representation, kernel, targets, and ridge setting fixed within each method, query-local top-k KRR lowers mean test RMSE relative to global KRR in all 27 corrected dataset–sample-size–kernel cells. All 27 exploratory paired bootstrap intervals favor local inference, and 24 of 27 cells favor it in all five seeds. Architecturally, the learned geometry first retrieves a small set of cases relevant to the query; the ridge solve then fits a query-specific predictor on that set rather than allowing globally dissimilar references to interfere. The frozen-model reference-bank experiment provides direct evidence for one part of this account: grossly corrupted stored inputs are almost always excluded from top-k for both RBF and Cauchy variants. The selected cases are also inspectable, making the prediction's reference set explicit.

This result is a within-method local-versus-global comparison, not a claim that local KRR universally beats Direct MLP or Gradient Boosting; those rankings remain dataset dependent. The label-noise screen covers only two mechanisms at one severity; it does not cover adversarial labels, adversarial queries, or arbitrary fixed input-space kernels. Exact inference still requires an $O(N)$ reference scan and an $O(k^3)$ local solve, so the statistical and interpretability benefits come with a deployment cost.

What remains empirical. Positive semidefiniteness is not evidence of predictive superiority, and the Hilbert embedding does not imply that Cauchy geometry aligns with a regression target better than RBF geometry. The fixed-scale Deep Cauchy control is needed to test whether input dependence in $\gamma(x)$ is useful; the matched Deep RBF control tests whether any gain is specific to Cauchy geometry. Direct and local/global comparisons provide context without changing those primary questions. In the completed five-seed artifacts, clean prediction, both input contamination severities, and the label-noise screen yield no stable Cauchy advantage.

Interpretation boundaries. The learned positive scale is not calibrated predictive uncertainty. Clean predictive parity did not settle robustness; the separate paired experiments also fail to show

a general Cauchy advantage. Likewise, an inspectable neighbor set is not a causal explanation, and discrete top-k selection is not differentiated end to end. The corresponding clarifications are in Appendix A.

Current empirical limitations. The `corrected-v3` candidate and robustness artifacts use three low-dimensional tabular datasets and five seeds. Their intervals are exploratory and do not establish superiority or equivalence. Finite hyperparameter budgets, fixed-kernel grid boundaries, dirty-arm preprocessing adaptation, and exact-search costs limit the scope of the empirical conclusions. The reference-bank isolation supports rejection of gross stored-input errors; the label screen supports conditional containment of target swaps, not resistance to arbitrary or adversarial label noise, adversarial queries, or calibrated uncertainty. The separate classification evidence is also limited to two common datasets and five seeds, with no confirmatory power analysis, probability calibration, or claim of statistical significance.

7 Conclusion

We introduced a learnable location–scale Cauchy–KL kernel for regression and established its positive-semidefinite validity, fixed-scale reduction, local pullback geometry, and differentiability through KRR. The construction gives a mathematically explicit way to learn both location and scale in similarity space. Its strongest positive empirical conclusion is that, conditional on a learned kernel geometry, query-local top-k KRR consistently improves over the same model's global-KRR inference. The reference-bank isolation further shows that this local selection can prevent grossly corrupted stored inputs from entering the solve for both RBF and Cauchy. When labels rather than inputs are corrupted, local selection cannot reject the ordinary-looking references, but the local solve can limit the spread of gross label swaps; diffuse Gaussian target noise may instead benefit from global averaging. The additional Cauchy flexibility, however, is not supported as a general predictive or robustness advantage in the completed five-seed artifacts. The evidence therefore supports learned local reference selection—with dataset-dependent competitiveness and exact search costs—not universal dominance of local KRR or Cauchy geometry.

A Reader FAQ: architecture, scale, and evidence

This reader-facing appendix collects distinctions that are necessary for interpreting the experiment but would interrupt the paper's main scientific argument if repeated in full in the body.

A.1 Does "matched" mean that methods share trained weights?

No. Each neural method owns an independent backbone and independent heads and is trained separately. Matching means that the backbone and linear μ head have the same architecture, dimensions, and per-seed initial parameter values. No parameter is tied across methods after initialization. The executable contract is stated in Appendix I.2.

A.2 Why retain an explicit location head in Direct MLP?

Direct MLP computes $x \to h_{\theta_a}(x) \to \mu_a(x) \to \widehat{y}$. Its two linear maps after the nonlinear backbone could be collapsed algebraically into one affine map. We retain μ_a explicitly so that all neural methods preserve the same perception-to-μ structure as closely as possible and diverge only after

that common structural boundary. The purpose is experimental matching, not an increase in Direct MLP expressivity.

A.3 Are the methods matched in total parameter count?

No. The controlled boundary is the perception-to-μ path. Deep Cauchy intentionally adds the positive $\gamma(x)$ branch whose value is being tested, while Direct MLP adds a scalar output and the kernel methods add their KRR prediction rule. The paper therefore says "perception-to-μ matched," not "fully parameter matched."

A.4 What is fixed in Fixed-scale Deep Cauchy?

The backbone and $\mu(x)$ remain learned. Only the Cauchy scale is constrained to one positive, learned scalar ℓ shared across inputs and dimensions: $\gamma_j(x) \equiv \ell$. Thus "fixed-scale" means input-invariant, not non-trainable. Comparing this model with Deep Cauchy isolates the contribution of input-dependent scale.

A.5 Is the learned scale predictive uncertainty?

No. In this model $\gamma(x)$ controls similarity geometry: it changes how movement in learned location and scale is penalized by the Cauchy–KL kernel. Without a separate probabilistic calibration result, it must not be described as predictive uncertainty. The local quadratic form in Appendix F gives its precise geometric interpretation.

A.6 How do N, B, and k differ?

N is the full training/reference bank, B is the number of examples in one differentiable KRR training solve, and k is the number of references in one query-local inference solve. Equal numerical values do not make B and k the same object. In the `corrected-v3` reference protocol, deep-kernel training uses $B = N$ and no top-k operation; top-k is applied only after training.

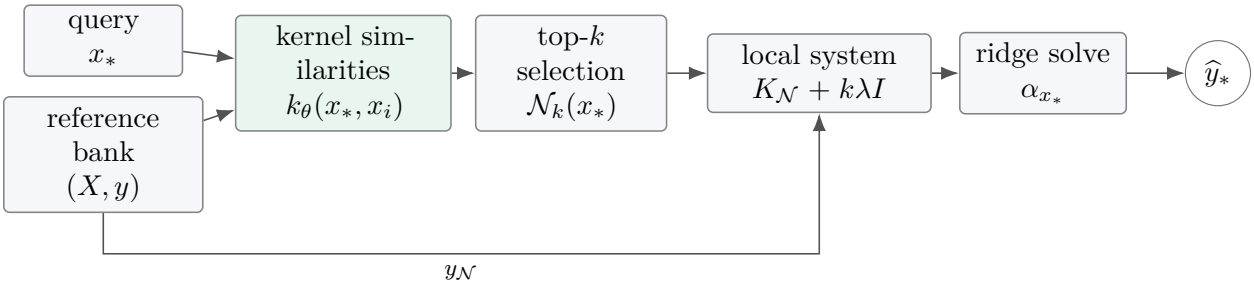

Figure 2: Query-local KRR after neural training. Each frozen query is compared with all N references, top-k chooses a subset, and a new ridge system is solved on that subset. The training-solve size B does not appear in this inference computation.

A.7 Is local top-k selection end-to-end differentiable?

Training and inference use the same learned kernel but not the same computation graph. The current deep-kernel trainer constructs one full $N \times N$ Gram matrix per epoch ($B = N$), solves against all training targets, and backpropagates that in-sample objective. It has no top-k retrieval

and no separate query/support pools. Direct MLP instead uses ordinary shuffled mini-batch scalar MSE. After training, neural parameters are frozen; only then does each inference query score all N references, select top-k, and solve a new local KRR system. Batching several inference queries is merely a memory-management device and does not shrink the reference bank.

The independent `classification-v1-exploratory` runner uses the same full-training-bank computation for its deep kernels, with a multiclass centered-simplex target matrix in place of the regression target vector. After each optimizer update it recomputes the training Gram matrix and ridge solution, then evaluates validation-to-training cross-kernel scores. It has no episodic support/query split. Its Direct classifier instead uses shuffled mini-batch cross-entropy.

Table 4: Training, validation, and inference are distinct computations in the `corrected-v3` protocol. The top-k operation belongs only to query-local inference.

Stage	Data participating	Operation and gradient status
Deep-kernel training	Full training partition, $B = N$	Construct $K_\theta(X_{\text{tr}}, X_{\text{tr}})$, solve for α, and backpropagate the in-sample MSE through the kernel and solve. No top-k.
Direct-MLP training	One shuffled scalar-MSE mini-batch	Predict through the scalar head and backpropagate its batch MSE. No kernel solve and no top-k.
Validation	Full training partition plus validation queries	Recompute the full training Gram matrix and α at the current parameter state, then evaluate validation MSE. No gradient and no top-k.
Query-local inference	Each query plus all N frozen training references	Score the query against all references, select top-k, construct the local Gram system, and solve for the prediction. Neural parameters are frozen.

A future batch-internal deep-KRR variant with $B < N$ would construct K_B, solve against targets from that same training batch, and compute its loss within that batch. Introducing distinct support and query pools would instead define a new episodic objective and require a new protocol. For a fixed inference neighborhood, kernel values and the ridge solve are differentiable; neighbor identities change discretely at similarity ties. Selected neighbors are inspectable reference cases, but neither selection nor KRR weights are causal explanations.

A.8 Which numerical evidence is current?

The `legacy-v0` archive has preprocessing leakage and a state-inconsistent validation solve. The later `corrected-v2` matrix repairs those defects but its Direct MLP omits the explicit linear μ head, so it cannot support the current Direct-versus-KRR comparison. Only `corrected-v3` implements the full perception-to-μ contract. A complete 315-row `corrected-v3` candidate now passes its structural and provenance gates. For the three learned kernels, local KRR lowers mean RMSE in all 27 dataset–sample-size–kernel cells, and all 27 exploratory paired-bootstrap intervals favor local over global inference. It shows no general Deep Cauchy advantage over Deep RBF, shared-scale Cauchy, Direct MLP, or the tree control. Separate 10% and nested 20% dirty-development screens also show no stable Cauchy-specific robustness. A separate 20% target-noise screen likewise shows no stable Cauchy-specific advantage: local KRR contains label-swap degradation more than global KRR in all nine learned-kernel cells, but not diffuse Gaussian target noise. All five-seed intervals remain exploratory. The historical JSON field named `xgboost` contains sklearn's

`GradientBoostingRegressor`, not the XGBoost library; current artifacts use `gradient_boosting`.

Classification evidence is kept under a separate protocol boundary. The 70-row exploratory classification screen covers two datasets, five paired seeds, and seven methods and passes its own structural acceptance gate. It does not enter the 315-row `corrected-v3` regression gate. Its tree baseline is sklearn's `GradientBoostingClassifier`, not XGBoost; its numerical results and limitations are summarized in Appendix C.7.

A.9 Does clean-data parity settle robustness?

No. Clean predictive accuracy and robustness to contaminated development data are separate hypotheses. A robustness claim requires a predeclared paired experiment that trains and selects models under matched corruption and evaluates them on the same untouched test set. The completed 10% and nested 20% screens satisfy that design but do not show a stable Cauchy advantage. A separate frozen-model reference-bank isolation shows a narrower mechanism: grossly corrupted stored references leave query-local top-k for both RBF and Cauchy, with little test-RMSE change. This is reference rejection, not evidence that $\gamma(x)$ is uncertainty or that Cauchy is universally robust. The target-noise screen exposes a different boundary: noisy labels stay in top-k, yet the local solve can contain gross label swaps; global inference can average diffuse zero-mean target noise more effectively.

A.10 Does multiclass classification require a second optimizer?

No, if the final predictor is kernel ridge classification. A C-column target matrix supplies C right-hand sides to the same regularized Gram system. The actual classification runner uses centered-simplex targets—one for the true class and $-1/(C-1)$ otherwise—rather than one-hot targets. One matrix factorization still returns all C class-score functions. During deep-kernel training this solve is a differentiable forward operation, not a separately trained inner optimizer; the persistent optimizer updates only the neural kernel parameters. The non-differentiable argmax is used only to decode validation or inference scores. The full calculation is in Appendix C.

B Notation and mathematical symbols

This section consolidates the notation used in the model, KRR solver, and complexity discussion. A plain integer k denotes a neighborhood size, whereas $k_\theta(x, x')$ denotes a kernel function. Likewise, n is the sample size in the mathematical derivation or benchmark cell, while N is the full training/reference-bank size in the scaling protocol.

Table 5: Mathematical symbols, their domains or shapes, and their roles in the paper.

Symbol	Domain or shape	Meaning and role
Data, indices, and sample counts		
i	$\{1, \dots, n\}$	Observation index.
j	$\{1, \dots, p\}$	Coordinate index in the learned kernel-parameter space.
$x_i,\ y_i$	$x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$	Input and scalar regression target for observation i.
c_i	$\{1, \dots, C\}$	Class label for observation i in a C-class problem.
$X,\ y$	$X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$	Input matrix and target vector used in a global KRR system.

Table 5 (continued)

Symbol	Domain or shape	Meaning and role
Y	$\mathbb{R}^{q\times C}$	One-hot or otherwise numerically encoded class-label matrix for a q-example multiclass kernel solve.
x_*	$\mathbb{R}^d$	Query input whose target is to be predicted.
n	$\mathbb{N}$	Number of observations in the current mathematical derivation or sampled benchmark cell.
N	$\mathbb{N}$	Total training/reference-bank size retained across training steps and available to exact neighbor search.
B	$1 \le B \le N$	Number of examples in one differentiable KRR training solve; `corrected-v3` uses $B = N$, while $B < N$ is prospective.
k	$1 \le k \le N$	Number of reference examples selected for one query-local inference solve; it is independent of B.
m	$\mathbb{N}$	Generic finite Gram-matrix or KRR-system size in theorem statements.
Learned representation and Cauchy parameters		
$d,\ r,\ p$	$\mathbb{N}$	Input dimension, backbone-output dimension, and kernel-parameter dimension, respectively.
θ_a	model parameters	Trainable parameters of neural method a; different methods own independent parameter instances.
h_{θ_a}	$\mathbb{R}^d \to \mathbb{R}^r$	Architecture-matched perception/backbone map for neural method a.
$z_a = h_{\theta_a}(x)$	$\mathbb{R}^r$	Learned hidden representation of input x for method a.
$\mu_a(x)$	$\mathbb{R}^p$	Linear hidden/location vector used by Direct MLP, Deep RBF, and both Deep Cauchy variants.
$w_{o,a},\ b_{o,a}$	$w_{o,a} \in \mathbb{R}^p$, $b_{o,a} \in \mathbb{R}$	Direct MLP scalar output layer applied after $\mu_a(x)$.
$\gamma_a(x)$	$\mathbb{R}^p_{>0}$	Positive, input-dependent Cauchy scale vector available only to Deep Cauchy. It is not assumed to be calibrated uncertainty.
ε	$\mathbb{R}_{>0}$	Positive numerical floor added after softplus to keep every Cauchy scale strictly positive.
P_x	probability law	Product distribution $\bigotimes_{j=1}^{p} \mathrm{Cauchy}(\mu_j(x), \gamma_j(x))$ associated with input x.
Divergences and kernels		
$D_{\mathrm{KL}}(P\|Q)$	$\mathbb{R}_{\ge 0}$	Kullback–Leibler divergence from distribution P to Q. On the univariate Cauchy location-scale family used here, its closed form is symmetric.
d_{C}	$\mathbb{R}_{\ge 0}$	One-coordinate Cauchy KL divergence between two location-scale parameter pairs.
$D_{\mathrm{C}}(x, x')$	$\mathbb{R}_{\ge 0}$	Sum of d_{C} over p coordinates; equivalently $D_{\mathrm{KL}}(P_x\|P_{x'})$.
$k_\theta(x, x')$	$[0, 1]$	Generic learned kernel similarity; the subscript distinguishes the function from the integer neighborhood size k.
k_{RBF}	$(0, 1]$	RBF kernel applied to learned locations $\mu_\theta(x)$.
k_{C}	$(0, 1]$	Product Cauchy-KL kernel $\exp(-\beta D_{\mathrm{C}})$.
ℓ	$\mathbb{R}_{>0}$	RBF length scale; also the one shared scale in the fixed-scale Cauchy ablation when stated explicitly.
β	$\mathbb{R}_{>0}$	Exponent controlling the decay of Cauchy-KL kernel similarity.

Table 5 (continued)

Symbol	Domain or shape	Meaning and role
Kernel ridge regression and local inference		
K	$\mathbb{R}^{q\times q}$	Gram matrix for the q examples in the current global, batch, or local KRR solve.
$K_{\mathcal{N}}$	$\mathbb{R}^{k\times k}$	Gram matrix restricted to the selected query neighborhood.
I_q	$\mathbb{R}^{q\times q}$	Identity matrix matching the current KRR-system size q.
λ	$\mathbb{R}_{>0}$	Ridge parameter in the paper's size-normalized system $K + q\lambda I_q$.
α	$\mathbb{R}^q$	KRR coefficient vector obtained from the regularized linear solve.
A or A_T	$\mathbb{R}^{q\times C}$	Multiclass kernel-ridge coefficient matrix, with one coefficient column per class.
k_*	$\mathbb{R}^q$	Kernel-similarity vector between query x_* and the q reference examples in the current solve.
$\mathcal{N}_k(x_*)$	subset of $\{1,\dots,N\}$	Indices of the k references with largest similarity to query x_*.
$\widehat{f}(x_*)$	$\mathbb{R}$	Predicted target, computed as a kernel vector times the corresponding KRR coefficients.
$s(x_*)$	$\mathbb{R}^C$	Multiclass score vector before decoding or probability calibration.
$\widehat{c}(x_*)$	$\{1,\dots,C\}$	Predicted class, usually obtained by an argmax over class scores.
e_c	$\{0,1\}^C$	One-hot vector for class c.
E	$\{0,1\}^{q\times C}$	One-hot class-target matrix used to state the fixed-system equivalence.
T	$\mathbb{R}^{q\times C}$	Centered-simplex class-target matrix used by the classification runner.
$\Delta y,\ \Delta T$	$\mathbb{R}^q,\ \mathbb{R}^{q\times C}$	Regression-target or centered-simplex class-target perturbation used to trace the direct effect of label corruption.

C Kernel prediction primer: regression and classification

The primary paper evaluates regression, with a separate exploratory classification screen in Appendix C.7. The Cauchy–KL construction is a kernel, not a regression-only prediction rule. This section separates three objects that are easy to conflate: the kernel defines similarity, the training loss defines what is fitted, and an output decoder turns the fitted function into a number, class, or probability. Positive semidefiniteness makes the same kernel admissible in each construction below; it does not make their losses or statistical interpretations identical.

C.1 Kernel ridge regression from the RKHS objective

Let k be a positive-semidefinite kernel with reproducing-kernel Hilbert space $\mathcal{H}_k$. Given q examples $\{(x_i, y_i)\}_{i=1}^q$, kernel ridge regression solves

$$\min_{f\in\mathcal{H}_k} \frac{1}{q}\sum_{i=1}^{q}\big(y_i - f(x_i)\big)^2 + \lambda\|f\|_{\mathcal{H}_k}^2, \qquad \lambda > 0. \tag{19}$$

The representer theorem implies that a minimizer has the finite expansion

$$f(\cdot) = \sum_{i=1}^{q} \alpha_i k(\cdot, x_i). \tag{20}$$

Writing $K_{ij} = k(x_i, x_j)$ gives the coefficient objective

$$\frac{1}{q}\|y - K\alpha\|_2^2 + \lambda \alpha^\top K \alpha. \tag{21}$$

Its canonical ridge solution is

$$\alpha = (K + q\lambda I_q)^{-1} y, \qquad \widehat{f}(x_*) = k_*^\top \alpha, \qquad (k_*)_i = k(x_*, x_i). \tag{22}$$

Because $K \succeq 0$ and $\lambda > 0$, $K + q\lambda I_q$ is positive definite and invertible. Thus regression prediction is a similarity-weighted linear combination of training targets after the similarities have been globally coupled and regularized by the inverse Gram system. It is generally not a simple normalized kernel average.

C.2 Query-local KRR

For a query x_*, let $\mathcal{N} = \mathcal{N}_k(x_*)$ contain the k references with the largest learned-kernel similarity. Query-local KRR replaces the q-example system in Eq. (22) by

$$\alpha_*(x_*) = (K_{\mathcal{N}} + k\lambda I_k)^{-1} y_{\mathcal{N}}, \qquad \widehat{f}_{\text{local}}(x_*) = k_{*,\mathcal{N}}^\top \alpha_*(x_*). \tag{23}$$

The coefficients therefore depend on the query: changing x_* can change both the selected set and the local linear system. This is the precise sense in which inference is KNN-like, but it is not ordinary KNN regression. KNN usually averages selected targets directly; local KRR fits a regularized kernel predictor inside the selected neighborhood. Local learning of this form is classical [2]; the learned geometry and differentiable training solve are the paper-specific ingredients.

C.3 Binary classification by kernel ridge classification

For binary labels encoded as $y_i \in \{-1, +1\}$, the same squared-loss problem and closed-form solve produce a real-valued decision score. The classifier is

$$\widehat{c}(x_*) = \begin{cases} +1, & k_*^\top (K + q\lambda I_q)^{-1} y \geq 0, \\ -1, & \text{otherwise.} \end{cases} \tag{24}$$

This construction is often called kernel ridge classification or least-squares classification. The magnitude of the score is a margin-like quantity, not a calibrated class probability.

C.4 Multiclass kernel ridge classification

For C classes, let $E \in \mathbb{R}^{q \times C}$ be the one-hot matrix, $E_{ic} = \mathbf{1}\{c = c_i\}$. The classification runner instead uses the centered-simplex target matrix $T \in \mathbb{R}^{q \times C}$,

$$T_{ic} = \begin{cases} 1, & c = c_i, \\ -1/(C-1), & c \neq c_i, \end{cases} \qquad T = \frac{C}{C-1} E - \frac{1}{C-1} \mathbf{1}_q \mathbf{1}_C^\top. \tag{25}$$

Each row of T sums to zero, and the binary case reduces to targets $+1$ and -1. Squared loss still separates over the C output columns, so all class functions are obtained in one solve:

$$A_T = (K + q\lambda I_q)^{-1} T, \qquad s_T(x_*) = k_*^\top A_T, \qquad \widehat{c}(x_*) = \arg\max_{c \in \{1,\dots,C\}} s_{T,c}(x_*). \tag{26}$$

There is a useful but limited relation to one-hot coding. Fix the kernel, ridge value, and reference set, write $M = K + q\lambda I_q$, and let $s_E(x_*) = k_*^\top M^{-1} E$. Then

$$s_T(x_*) = \frac{C}{C-1} s_E(x_*) - \frac{1}{C-1} \underbrace{k_*^\top M^{-1} \mathbf{1}_q}_{\rho(x_*)} \mathbf{1}_C^\top. \tag{27}$$

Thus centered-simplex and one-hot scores differ only by a positive scale and a query-dependent offset shared by every class, so their argmax is identical under those fixed quantities. This does *not* make their training MSE, MSE-based hyperparameter selection, raw score magnitudes, or probability interpretation equivalent. In particular, changing the target coding while learning the kernel can change the fitted representation.

To see why all scores remain available in closed form, write $T = [t^{(1)}, \dots, t^{(C)}]$ and $A_T = [a^{(1)}, \dots, a^{(C)}]$. The Frobenius loss is a sum of C ridge objectives sharing the same Gram matrix, so

$$a^{(c)} = (K + q\lambda I_q)^{-1} t^{(c)}, \qquad c = 1, \dots, C. \tag{28}$$

One matrix factorization is therefore reused for all C right-hand sides.

The vector $s(x_*)$ contains class scores. Applying a softmax to these scores produces numbers that sum to one, but does not by itself make them calibrated probabilities; calibration requires an additional validated procedure or a probabilistic training loss.

Why this remains end-to-end. The actual `classification-v1-exploratory` runner does not construct episodes. With all N training examples in each differentiable solve, it computes

$$M_\theta = K_\theta(X_{\mathrm{tr}}, X_{\mathrm{tr}}) + N\lambda_\theta I_N, \quad A_\theta = M_\theta^{-1} T_{\mathrm{tr}}, \quad S_{\mathrm{tr},\theta} = K_\theta(X_{\mathrm{tr}}, X_{\mathrm{tr}}) A_\theta, \tag{29}$$

and minimizes the in-sample centered-simplex objective $\|T_{\mathrm{tr}} - S_{\mathrm{tr},\theta}\|_{\mathrm{F}}^2/(NC)$. After every optimizer update, validation recomputes M_θ and A_θ at that same parameter state and evaluates

$$S_{\mathrm{val},\theta} = K_\theta(X_{\mathrm{val}}, X_{\mathrm{tr}}) A_\theta \tag{30}$$

against T_{val}. Thus validation uses a freshly recomputed cross-kernel; it does not reuse stale coefficients. There is one persistent optimizer for the neural and kernel parameters, while A_θ is recomputed inside the forward pass rather than optimized as a separate parameter. Differentiating the solve gives

$$\mathrm{d}A_\theta = -M_\theta^{-1}(\mathrm{d}M_\theta) A_\theta, \qquad \mathrm{d}S_{\mathrm{tr},\theta} = (\mathrm{d}K_{\mathrm{tr,tr},\theta}) A_\theta + K_{\mathrm{tr,tr},\theta} \mathrm{d}A_\theta. \tag{31}$$

The argmax in Eq. (26) is excluded from this training graph and is used only to decode scores. A distinct support/query split could reduce the role of self-similarity, but it would define a new episodic protocol; it is not part of this experiment.

The query-local multiclass analogue is immediate:

$$A_*(x_*) = (K_{\mathcal{N}} + k\lambda I_k)^{-1} T_{\mathcal{N}}, \quad s_{\mathrm{local}}(x_*) = k_{*,\mathcal{N}}^\top A_*(x_*), \quad \widehat{c}_{\mathrm{local}}(x_*) = \arg\max_c s_{\mathrm{local},c}(x_*). \tag{32}$$

A small local neighborhood can contain no examples of a rare class. Practical classification variants should therefore validate k and may use an adaptive neighborhood, a class prior, or a global fallback.

C.5 Changing the loss: kernel logistic regression and SVMs

Classification does not require squared loss. In multiclass kernel logistic regression, each logit has the representer form $f_c(\cdot) = \sum_i a_{ic} k(\cdot, x_i)$ and the coefficients minimize

$$-\frac{1}{q}\sum_{i=1}^{q} \log \frac{\exp f_{c_i}(x_i)}{\sum_{r=1}^{C} \exp f_r(x_i)} + \frac{\lambda}{2}\sum_{c=1}^{C} \|f_c\|_{\mathcal{H}_k}^2. \tag{33}$$

Prediction uses softmax probabilities and their argmax. Unlike KRR, Eq. (33) has no general closed-form coefficient solution and is optimized iteratively [20].

For a binary support-vector machine, squared loss is replaced by hinge loss:

$$\min_{f \in \mathcal{H}_k} \frac{1}{q}\sum_{i=1}^{q} \max\{0, 1 - y_i f(x_i)\} + \frac{\lambda}{2}\|f\|_{\mathcal{H}_k}^2, \qquad \widehat{c}(x_*) = \operatorname{sign} f(x_*). \tag{34}$$

Only support vectors have nonzero dual coefficients [4]. Kernel ridge classification, kernel logistic regression, and a kernel SVM can therefore use exactly the same Cauchy–KL or RBF Gram matrix while producing different coefficients, probability interpretations, and decision boundaries.

C.6 What label corruption changes

With a frozen kernel and frozen reference set, the direct label-noise channel in KRR is exactly linear. A regression-target perturbation Δy changes the global prediction by

$$\Delta \widehat{f}_{\text{global}}(x_*) = k_*^\top (K + q\lambda I_q)^{-1} \Delta y, \tag{35}$$

whereas a fixed local neighborhood gives

$$\Delta \widehat{f}_{\text{local}}(x_*) = k_{*,\mathcal{N}}^\top (K_{\mathcal{N}} + k\lambda I_k)^{-1} \Delta y_{\mathcal{N}}. \tag{36}$$

For multiclass ridge classification the corresponding score perturbation is

$$\Delta s_{\text{global}}(x_*) = k_*^\top (K + q\lambda I_q)^{-1} \Delta T, \qquad \Delta s_{\text{local}}(x_*) = k_{*,\mathcal{N}}^\top (K_{\mathcal{N}} + k\lambda I_k)^{-1} \Delta T_{\mathcal{N}}. \tag{37}$$

Under Eq. (25), relabeling one example from class a to class b changes one row by $C(e_b - e_a)/(C-1)$. If that reference is outside $\mathcal{N}_k(x_*)$, its direct contribution to the local score perturbation is exactly zero; if it is inside the neighborhood, local selection does not remove it. More generally,

$$\|\Delta s(x_*)\|_2 \le \left\| k_*^\top (K + q\lambda I_q)^{-1} \right\|_2 \|\Delta T\|_{\mathrm{F}}, \tag{38}$$

with the analogous restricted bound for a local solve. Neither bound orders local and global inference universally: localization can contain a gross mismatch, while a global solve can average diffuse zero-mean noise.

The completed label-noise experiment retrains the neural representation under dirty labels. It therefore also contains an indirect geometry channel through the learned parameters θ; Eqs. (35) to (37) isolate only the direct propagation that remains after the kernel is fixed. This distinction explains why label-swap and Gaussian target noise need not produce the same local-versus-global ordering.

C.7 Exploratory classification evidence

The independent `classification-v1-exploratory` screen aligns the neural perception structure while preserving each method's prediction rule:

- Direct MLP uses backbone $\rightarrow$ linear $\mu \rightarrow C$ logits and is trained with cross-entropy;
- Deep RBF uses backbone $\rightarrow$ linear $\mu \rightarrow$ RBF kernel ridge classification;
- Deep Cauchy uses backbone $\rightarrow$ linear μ and positive γ heads $\rightarrow$ Cauchy–KL kernel ridge classification; and
- Fixed-scale Deep Cauchy uses backbone $\rightarrow$ linear μ plus one positive, trainable scale shared by all inputs $\rightarrow$ Cauchy–KL kernel ridge classification.

For a given seed, the shared backbone and μ prefix start from bitwise identical parameters. The four methods are nevertheless instantiated and trained independently; "shared" describes the architecture and seeded initialization contract, not tied learned weights. The two fixed kernels have no neural perception network. The tree baseline is sklearn's `GradientBoostingClassifier`.

The screen uses stratified 60/20/20 splits of Breast Cancer and Digits with train-only standardization, five paired seeds (42–46), and seven methods. Its acceptance gate requires exactly $2 \times 5 \times 7 = 70$ unique dataset–seed–method rows, the declared protocol and target coding, paired split hashes, finite metrics, the neural-prefix contract, the real estimator class, and fixed-grid boundary audits. The completed artifact passes that gate independently of the `corrected-v3` regression matrix.

Table 6: Exploratory classification results across five paired seeds (mean $\pm$ standard deviation). Kernel entries use query-local $k = 100$ inference. Higher is better for both reported metrics.

Method	Breast Cancer balanced accuracy	Digits macro-F1
Gradient Boosting classifier	$.9596 \pm .0178$	$.9513 \pm .0081$
Direct MLP	$.9696 \pm .0104$	$.9655 \pm .0075$
Fixed RBF (local)	$.9645 \pm .0096$	$.9800 \pm .0051$
Fixed Cauchy (local)	$.9603 \pm .0145$	$.9894 \pm .0037$
Deep RBF (local)	$.9667 \pm .0220$	$.9762 \pm .0082$
Deep Cauchy (local)	$.9690 \pm .0174$	$.9785 \pm .0024$
Fixed-scale Deep Cauchy (local)	$.9667 \pm .0180$	$.9762 \pm .0086$

Using higher-is-better paired differences, Breast Cancer gives Deep Cauchy minus Deep RBF $+.0024$ (wins/ties/losses 1/4/0) and Deep Cauchy minus its fixed-scale deep control $+.0023$ (1/4/0). Digits gives Deep Cauchy minus Deep RBF $+.0023$ (3/0/2), while Deep Cauchy minus the fixed input-space Cauchy kernel is $-.0110$ (0/0/5). Local-minus-global gains for the kernel methods are generally small on both datasets. These five-seed summaries are an exploratory stability screen: they support neither a significance claim, nor universal classification superiority, nor a probability-calibration claim. In particular, they show no stable benefit from input-dependent Cauchy scale.

There are also material optimization-budget boundaries. On Digits, all 15 rows from the three deep-kernel methods executed the full 80-epoch cap. The runner restored the best validation checkpoint observed within those 80 epochs, but the artifact did not record the best epoch or learning curves; these rows therefore must not be described as converged.

Fixed Cauchy searched 350 hyperparameter candidates per seed, whereas each deep method used one declared configuration and one initialization per seed. The Gradient Boosting classifier searched only four candidates, and all 10 dataset–seed rows chose the upper tested learning-rate boundary, 0.1. It is consequently a declared sklearn baseline, not evidence for a sufficiently tuned boosting optimum. These unequal budgets further limit cross-method ranking claims.

A post-hoc Digits sensitivity run changed only the deep methods' maximum epoch budget from 80 to 200, retaining the same seeds, learning rate, patience, and architecture. It is not part of the canonical 70-row table. Local macro-F1 became $.9806 \pm .0035$ for Deep RBF, $.9801 \pm .0060$ for Deep Cauchy, and $.9790 \pm .0052$ for fixed-scale Deep Cauchy. Deep Cauchy minus Deep RBF changed from $+.0023$ (3/0/2) to $-.0005$ (1/0/4); Deep Cauchy minus its fixed-scale deep control was $+.0011$ (3/0/2), and Deep Cauchy minus fixed input-space Cauchy remained negative at $-.0094$ (1/0/4). Several best checkpoints occurred at or near epoch 200, so this sensitivity run also does not establish convergence. Its proper interpretation is that the small Cauchy-versus-RBF ordering is sensitive to the training budget, further weakening any stable Cauchy-advantage claim.

C.8 Scope of the classification extension

The validity results for the Cauchy–KL Gram matrix carry over immediately to all of the kernel predictors above. The paper's primary confirmatory target remains scalar regression, and `classification-v1-exploratory` is a separate two-dataset screen rather than an extension of the `corrected-v3` regression gate. Equations Eqs. (24), (26), (33) and (34) distinguish several valid classification predictors; only centered-simplex kernel ridge classification and the matched Direct/tree controls were tested in Table 6. Their KRR outputs are decision scores, not calibrated probabilities.

D Cauchy KL divergence in closed form

This section derives Eq. (10) rather than treating it as a kernel formula to be accepted on inspection.

Lemma 3 (Poisson integral identity)**.** *If $X \sim \mathrm{Cauchy}(\mu, \gamma)$ with $\gamma > 0$, then for every $\nu \in \mathbb{R}$ and $\eta > 0$,*

$$\mathbb{E}\Big[\log\left((X-\nu)^2+\eta^2\right)\Big] = \log\left((\mu-\nu)^2+(\gamma+\eta)^2\right). \tag{39}$$

Proof. The Cauchy density is the Poisson kernel of the upper half-plane:

$$p_{\mu,\gamma}(x) = \frac{1}{\pi}\frac{\gamma}{(x-\mu)^2+\gamma^2}. \tag{40}$$

Let $z = \mu+i\gamma$ and $a = \nu-i\eta$. Because a lies in the lower half-plane, the function $u(z) = 2\log|z-a|$ is harmonic throughout the upper half-plane. Its boundary value on $x \in \mathbb{R}$ is $u(x) = \log((x-\nu)^2+\eta^2)$. The Poisson representation therefore gives

$$\begin{aligned}\int_{\mathbb{R}} p_{\mu,\gamma}(x)\log\left((x-\nu)^2+\eta^2\right)dx &= u(\mu+i\gamma) &(41)\\ &= 2\log|\mu+i\gamma-(\nu-i\eta)| &(42)\\ &= \log\left((\mu-\nu)^2+(\gamma+\eta)^2\right). &(43)\end{aligned}$$

The logarithmic boundary growth is integrable against the $O(x^{-2})$ Poisson kernel; equivalently, the identity follows first for truncated boundary data and then by dominated convergence. □

Proposition 4 (Univariate Cauchy KL)**.** *For $P = \mathrm{Cauchy}(\mu, \gamma)$ and $Q = \mathrm{Cauchy}(\nu, \eta)$,*

$$D_{\mathrm{KL}}(P\|Q) = \log \frac{(\mu-\nu)^2 + (\gamma+\eta)^2}{4\gamma\eta}. \tag{44}$$

In particular, the divergence is symmetric on this family and equals zero if and only if $(\mu, \gamma) = (\nu, \eta)$.

Proof. The density ratio is

$$\log \frac{p_{\mu,\gamma}(x)}{p_{\nu,\eta}(x)} = \log \frac{\gamma}{\eta} + \log\left((x-\nu)^2 + \eta^2\right) - \log\left((x-\mu)^2 + \gamma^2\right). \tag{45}$$

Taking expectation under P and applying Lemma 3 twice yields

$$D_{\mathrm{KL}}(P\|Q) = \log \frac{\gamma}{\eta} + \log\left((\mu-\nu)^2 + (\gamma+\eta)^2\right) - \log(4\gamma^2) \tag{46}$$

$$= \log \frac{(\mu-\nu)^2 + (\gamma+\eta)^2}{4\gamma\eta}. \tag{47}$$

Symmetry is visible in the final expression. Its fraction is at least one because

$$(\mu-\nu)^2 + (\gamma+\eta)^2 - 4\gamma\eta = (\mu-\nu)^2 + (\gamma-\eta)^2 \geq 0, \tag{48}$$

with equality exactly when both parameters agree. □

Proposition 5 (Product additivity)**.** *Let $P = \bigotimes_{j=1}^{p} P_j$ and $Q = \bigotimes_{j=1}^{p} Q_j$, where every P_j is absolutely continuous with respect to Q_j and all component divergences are finite. Then*

$$D_{\mathrm{KL}}(P\|Q) = \sum_{j=1}^{p} D_{\mathrm{KL}}(P_j\|Q_j). \tag{49}$$

Proof. Writing $p = \prod_j p_j$ and $q = \prod_j q_j$,

$$D_{\mathrm{KL}}(P\|Q) = \int \prod_{r=1}^{p} p_r(x_r) \log \prod_{j=1}^{p} \frac{p_j(x_j)}{q_j(x_j)}\, dx_1 \cdots dx_p \tag{50}$$

$$= \sum_{j=1}^{p} \int p_j(x_j) \log \frac{p_j(x_j)}{q_j(x_j)}\, dx_j, \tag{51}$$

where integration over all coordinates $r \neq j$ equals one. This is the claimed sum. □

E Positive definiteness of the learned Cauchy kernel

We use the following result of Nielsen and Okamura [13, Theorem 11].

Lemma 6 (Hilbertian Cauchy KL)**.** *There are a real Hilbert space $\mathcal{H}$ and an injective map $\Phi : \mathbb{R} \times \mathbb{R}_{>0} \to \mathcal{H}$ such that*

$$d_{\mathrm{C}}(a, b) = \|\Phi(a) - \Phi(b)\|_{\mathcal{H}}^2 \tag{52}$$

for all univariate Cauchy parameter pairs a and b.

The cited result is nontrivial: it proves conditional negative definiteness of Cauchy KL using its hyperbolic-space representation. The remainder of the argument below is specific to our product and learned-map construction.

Proof of Theorem 1. For a product parameter tuple $a = (a_1, \dots, a_p)$, define the direct-sum embedding

$$\widetilde{\Phi}(a) = \Phi(a_1) \oplus \cdots \oplus \Phi(a_p) \in \mathcal{H}^{\oplus p}. \tag{53}$$

By Proposition 5 and Lemma 6,

$$D_{\mathrm{C}}(a, b) = \sum_{j=1}^{p} \|\Phi(a_j) - \Phi(b_j)\|_{\mathcal{H}}^2 = \|\widetilde{\Phi}(a) - \widetilde{\Phi}(b)\|_{\mathcal{H}^{\oplus p}}^2. \tag{54}$$

Hence D_{C} is conditionally negative definite. Schoenberg's theorem [16] implies that

$$\kappa(a, b) = \exp\big(-\beta D_{\mathrm{C}}(a, b)\big) \tag{55}$$

is positive semidefinite for every $\beta > 0$. Pulling a kernel back through an arbitrary function preserves positive semidefiniteness. Applying that fact to $a = \Psi_\theta(x) = (\mu_\theta(x), \gamma_\theta(x))$ proves the first claim.

For strictness, fix $x_1, \dots, x_m$ whose joint parameter tuples are distinct and put $u_i = \widetilde{\Phi}(\Psi_\theta(x_i))$. Injectivity of Φ makes the u_i distinct. They lie in a finite-dimensional affine subspace of $\mathcal{H}^{\oplus p}$, which can be identified isometrically with $\mathbb{R}^r$ for some $r \le m - 1$. For any nonzero $c \in \mathbb{R}^m$, the Gaussian Fourier identity gives

$$\sum_{i,j=1}^{m} c_i c_j e^{-\beta\|u_i - u_j\|^2} = C_{\beta,r} \int_{\mathbb{R}^r} \left| \sum_{i=1}^{m} c_i e^{i\omega^\top u_i} \right|^2 e^{-\|\omega\|^2/(4\beta)}\, d\omega, \tag{56}$$

where $C_{\beta,r} > 0$. If the right-hand side were zero, the finite exponential sum would vanish everywhere by continuity. Choose $v \in \mathbb{R}^r$ such that the projections $v^\top u_i$ are all distinct; such v exists outside a finite union of hyperplanes. Restricting to $\omega = tv$ would give

$$\sum_{i=1}^{m} c_i e^{itv^\top u_i} = 0 \quad \text{for every } t. \tag{57}$$

Differentiating at $t = 0$ for orders $0, \dots, m - 1$ yields a Vandermonde system with distinct nodes $v^\top u_i$, forcing every $c_i = 0$, a contradiction. Thus Eq. (56) is positive for nonzero c, and the Gram matrix is strictly positive definite. □

Remark 7 (Why injectivity is a condition, not an architecture claim). The validity result does not require an invertible backbone. Strict positive definiteness on a finite sample requires only that the final joint Cauchy parameter tuples be distinct on that sample. If two inputs collapse to the same tuple, their kernel rows coincide and the Gram matrix cannot be strictly positive definite. Ridge regularization still makes the KRR system invertible.

F Fixed scale and local geometry

Proof of Corollary 2. Substitute $\mu = x$, $\mu' = x'$, and $\gamma = \gamma' = \ell$ into Eq. (10):

$$d_{\mathrm{C}} = \log \frac{(x - x')^2 + 4\ell^2}{4\ell^2} = \log\left(1 + \frac{(x - x')^2}{4\ell^2}\right). \tag{58}$$

Exponentiation by $-\beta$ gives the stated kernel. □

Proposition 8 (Pullback quadratic form)**.** *Suppose every μ_j and γ_j is three times continuously differentiable near t, with $\gamma_j(t) > 0$. Then*

$$D_{\mathrm{C}}(t, t+\delta) = \frac{\delta^2}{4} \sum_{j=1}^{p} \frac{\dot{\mu}_j(t)^2 + \dot{\gamma}_j(t)^2}{\gamma_j(t)^2} + O(\delta^3). \tag{59}$$

Proof. It is enough to expand one component. Write $m = \mu(t)$, $g = \gamma(t)$, $a = \dot{\mu}(t)$, and $b = \dot{\gamma}(t)$. Taylor's theorem gives

$$\mu(t+\delta) = m + a\delta + O(\delta^2), \tag{60}$$
$$\gamma(t+\delta) = g + b\delta + \tfrac{1}{2}\ddot{\gamma}(t)\delta^2 + O(\delta^3). \tag{61}$$

The numerator and denominator inside Eq. (10) have the expansions

$$(\Delta\mu)^2 + (g+g')^2 = 4g^2 + 4gb\delta + \big(a^2 + b^2 + 2g\ddot{\gamma}(t)\big)\delta^2 + O(\delta^3), \tag{62}$$
$$4gg' = 4g^2 + 4gb\delta + 2g\ddot{\gamma}(t)\delta^2 + O(\delta^3). \tag{63}$$

Dividing cancels both the linear term and the second-derivative term:

$$\frac{(\Delta\mu)^2 + (g+g')^2}{4gg'} = 1 + \frac{a^2+b^2}{4g^2}\delta^2 + O(\delta^3). \tag{64}$$

Using $\log(1+u) = u + O(u^2)$ gives the component expansion. Summing components proves the result. □

The associated infinitesimal quadratic form is therefore

$$ds^2 = \frac{1}{4} \sum_{j=1}^{p} \frac{d\mu_j^2 + d\gamma_j^2}{\gamma_j^2}. \tag{65}$$

This is the pullback of a hyperbolic-type metric on the Cauchy parameter half-plane. Small scales magnify both location and scale changes. The result is a geometric interpretation of the learned scales, not a guarantee that the scales track the smoothness of the regression function.

G Gradients and differentiable KRR

G.1 Kernel derivatives

For one component, set

$$A = (\mu - \mu')^2 + (\gamma + \gamma')^2. \tag{66}$$

The derivatives of Eq. (10) are

$$\frac{\partial d_{\mathrm{C}}}{\partial \mu} = \frac{2(\mu-\mu')}{A}, \qquad \frac{\partial d_{\mathrm{C}}}{\partial \mu'} = -\frac{2(\mu-\mu')}{A}, \tag{67}$$
$$\frac{\partial d_{\mathrm{C}}}{\partial \gamma} = \frac{2(\gamma+\gamma')}{A} - \frac{1}{\gamma}, \qquad \frac{\partial d_{\mathrm{C}}}{\partial \gamma'} = \frac{2(\gamma+\gamma')}{A} - \frac{1}{\gamma'}. \tag{68}$$

Since $k_{\mathrm{C}} = e^{-\beta D_{\mathrm{C}}}$,

$$dk_{\mathrm{C}} = -k_{\mathrm{C}}\big(D_{\mathrm{C}}\, d\beta + \beta\, dD_{\mathrm{C}}\big). \tag{69}$$

The positive floor ε on the softplus scale head avoids division by zero; it is a numerical device, not part of the kernel theorem.

G.2 Differentiating the ridge solution

Let

$$A(\vartheta) = K(\vartheta) + n\lambda(\vartheta)I, \qquad \alpha(\vartheta) = A(\vartheta)^{-1}y. \tag{70}$$

Because $K \succeq 0$ and $\lambda > 0$, $A \succ 0$. Differentiating $A\alpha = y$ gives

$$d\alpha = -A^{-1}(dK + n\, d\lambda\, I)\alpha. \tag{71}$$

For a test kernel vector k_*,

$$d\widehat{f}(x_*) = (dk_*)^\top \alpha - k_*^\top A^{-1}(dK + n\, d\lambda\, I)\alpha. \tag{72}$$

Modern automatic-differentiation systems obtain the same expression through the linear solve without materializing a matrix inverse.

G.3 Local selection

For a fixed neighborhood $\mathcal{N}_k(x_*)$, the local formula is identical after replacing n by k and restricting K, y, and k_* to the neighborhood. The top-k map itself is piecewise constant and changes at similarity ties. Thus local inference is differentiable only within regions of parameter space that preserve the selected indices. The present implementation trains through full-batch global KRR and applies discrete top-k only at inference; it does not claim an end-to-end derivative through neighbor identity changes.

H Well-posedness and computational cost

Proposition 9 (Unique ridge solution)**.** *If $K \succeq 0$ and $\lambda > 0$, then $K + m\lambda I_m \succ 0$ and the KRR coefficient vector is unique.*

Proof. For every nonzero $v \in \mathbb{R}^m$,

$$v^\top(K + m\lambda I_m)v = v^\top Kv + m\lambda\|v\|_2^2 > 0. \tag{73}$$

Hence the matrix is positive definite and invertible. □

Let N denote the full training/reference bank, B the differentiable KRR training-solve size, and k the inference neighborhood. An exact global solve with $B = N$ costs $O(N^3)$ time and $O(N^2)$ memory. A batch solve costs $O(B^3)$ time and $O(B^2)$ memory; if a shuffled epoch visits all N examples once, approximately N/B solves give $O(NB^2)$ solve time per epoch. This accounting does not reduce the available data to B: all N rows can participate across steps and remain in the reference bank. The current `corrected-v3` deep-kernel runner uses $B = N$; the $B < N$ accounting describes a prospective extension rather than a completed reference result. Once a neighborhood is known, each local solve costs $O(k^3)$ time and $O(k^2)$ memory, independently of B. The current exact implementation evaluates query-to-training kernel similarities before top-k selection, so it also incurs an $O(Np)$ kernel-evaluation cost per query. Approximate neighbor indices could reduce that term for kernels whose search geometry admits a suitable index, but no such acceleration is part of the present evidence.

I Corrected protocol and reproducibility contract

I.1 Protocol defects and the later alignment mismatch

The `legacy-v0` archive must not be treated as reproduced evidence for two reasons:

1. **Preprocessing leakage.** The feature scaler was fitted before splitting, allowing validation and test covariates to influence the transformation applied to training data.

2. **State-inconsistent validation.** After an optimizer step, the validation kernel was evaluated at the new neural state while the KRR coefficient vector still came from the previous state.

The repaired loader splits raw data first and fits the feature transformation only on the training partition. The repaired trainer recomputes the training Gram matrix and ridge solution after every optimizer step before evaluating validation loss. Direct MLP additionally standardizes its scalar targets using training-partition statistics and reverses that transformation before RMSE; the deep-KRR rows use targets in their original units.

The later `corrected-v2` protocol fixed those two defects and matched the Direct MLP's backbone class and initialization. It did not, however, match the complete intended neural path: Direct attached its scalar output directly to the backbone, whereas the deep methods inserted a linear μ head. Its completed 315-row artifact is therefore diagnostic provenance, not a reference for the architecture-matched Direct-versus-KRR comparison. The mismatch is repaired only in `corrected-v3`.

I.2 Backbone-and-location-head matching contract

For each seed, all neural methods instantiate the same $d \to 128 \to 128 \to d$ ReLU backbone and the same $d \to d$ linear μ head in the same random-module construction order. Consequently, their backbone and μ parameters are equal at initialization. Each method nevertheless owns an independent parameter instance and optimizes it separately; no learned weights are tied across methods. The computation paths are:

Table 7: Exact neural matching boundary in `corrected-v3`. The common path is matched in architecture and initialization, while prediction rules and objectives remain method specific.

Method	Matched path	Method-specific continuation	Training target and objective
Direct MLP	$x \to h_{\theta_a}(x) \to \mu_a(x)$	scalar linear output $w_{o,a}^\top \mu_a + b_{o,a}$	train-standardized target; mini-batch scalar MSE
Deep RBF	$x \to h_{\theta_a}(x) \to \mu_a(x)$	RBF kernel on μ_a, then KRR	original-unit target; full-batch differentiable KRR
Deep Cauchy	$x \to h_{\theta_a}(x) \to \mu_a(x)$	parallel positive γ_a head, then Cauchy-KL KRR	original-unit target; full-batch differentiable KRR
Fixed-scale Deep Cauchy	$x \to h_{\theta_a}(x) \to \mu_a(x)$	one learned global scale ℓ, then Cauchy-KL KRR	original-unit target; full-batch differentiable KRR

The Direct path contains two consecutive linear maps after the nonlinear backbone. Although they can be collapsed into one affine map, μ_a remains explicit so that Direct MLP and the kernel

methods preserve the same perception-to-μ neural structure as closely as possible. This is the experimental matching boundary described in Appendix A.2. The Cauchy-only γ_a branch is an intentional mechanism parameter, so the models are perception-to-μ matched rather than equal in total parameter count. Objectives, batch regimes, target transformations, and inference rules remain method specific and are reported explicitly.

I.3 Matrix acceptance gate

A corrected reference artifact is accepted only if all of the following hold:

1. exactly 315 flat rows are present;
2. every combination of seven methods, three datasets, three sample sizes, and five seeds occurs exactly once;
3. every row declares protocol version `corrected-v3`;
4. every neural row declares the `backbone-linear-mu-v1` perception contract and the method-specific prediction path in Table 7;
5. every Direct MLP row declares its train-only target transformation;
6. no runner failure was converted into a successful partial matrix;
7. aggregate comparisons are paired by dataset, sample size, and seed;
8. the reviewed JSON is frozen before manuscript tables are regenerated.

The complete 315-row `corrected-v3` candidate has now executed with zero runner failures, finite metrics, the required architecture fields, and an independent representative verification. The earlier `corrected-v2` matrix remains diagnostic provenance and intentionally fails this gate because its Direct path omitted μ. Passing the gate makes the current comparisons auditable; it does not turn their heterogeneous five-seed results into empirical superiority.

References

[1] Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyxnZh0ct7.

[2] Léon Bottou and Vladimir Vapnik. Local learning algorithms. *Neural Computation*, 4(6): 888–900, 1992. doi: 10.1162/neco.1992.4.6.888.

[3] William S. Cleveland and Susan J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403): 596–610, 1988. doi: 10.1080/01621459.1988.10478639.

[4] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3): 273–297, 1995. doi: 10.1007/BF00994018.

[5] Jianqing Fan and Irene Gijbels. *Local Polynomial Modelling and Its Applications*. Chapman and Hall, 1996. doi: 10.1201/9780203748725.

[6] Jacob Goldberger, Sam Roweis, Geoffrey Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems*, volume 17, 2004.

[7] Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004. URL https://www.jmlr.org/papers/v5/jebara04a.html.

[8] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HklBjCEKvH.

[9] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019. doi: 10.1109/CVPR.2019.01091.

[10] Jean-Marc Mercier and Gabriele Santin. Differentiable kernel ridge regression for deep learning pipelines, 2026. URL https://arxiv.org/abs/2605.02313.

[11] Elizbar A. Nadaraya. On estimating regression. *Theory of Probability and Its Applications*, 9 (1):141–142, 1964. doi: 10.1137/1109020.

[12] Duy Nguyen-Tuong, Matthias Seeger, and Jan Peters. Model learning with local gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009. doi: 10.1163/016918609X12529286896877.

[13] Frank Nielsen and Kazuki Okamura. On f-divergences between cauchy distributions. *IEEE Transactions on Information Theory*, 2022. doi: 10.1109/TIT.2022.3231645. URL https://arxiv.org/abs/2101.12459.

[14] Christopher J. Paciorek and Mark J. Schervish. Nonstationary covariance functions for gaussian process regression. In *Advances in Neural Information Processing Systems*, volume 16, 2004.

[15] Massimiliano Patacchiola, Jack Turner, Elliot J. Crowley, Michael O'Boyle, and Amos Storkey. Bayesian meta-learning for the few-shot setting via deep kernels. In *Advances in Neural Information Processing Systems*, volume 33, pages 16108–16118, 2020.

[16] I. J. Schoenberg. Metric spaces and positive definite functions. *Transactions of the American Mathematical Society*, 44(3):522–536, 1938. doi: 10.2307/1989894.

[17] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[18] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009. URL https://www.jmlr.org/papers/v10/weinberger09a.html.

[19] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep kernel learning. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 370–378, 2016. URL https://proceedings.mlr.press/v51/wilson16.html.

[20] Ji Zhu and Trevor Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005. doi: 10.1198/106186005X25619.