Title
Detecting Label Noise via Leave-One-Out Cross-Validation

Authors
Tang, Yu-Hang
Zhu, Yuanran
Jong, Wibe A de

Publication Date
2021-03-21

Peer reviewed

Detecting Label Noise via Leave-One-Out Cross Validation

Yu-Hang Tang[1*], Yuanran Zhu[2], and Wibe A. de Jong[1]

[1]Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA
[2]University of California, Merced, Merced, California 94720, USA
[*]Correspondence: tang@lbl.gov

Abstract We present a simple algorithm for identifying and correcting real-valued noisy labels from a mixture of clean and corrupted samples using Gaussian process regression. A heteroscedastic noise model is employed, in which additive Gaussian noise terms with independent variances are associated with each and all of the observed labels. Thus, the method effectively applies a sample-specific Tikhonov regularization term, generalizing the uniform regularization prevalent in standard Gaussian process regression. Optimizing the noise model using maximum likelihood estimation leads to the containment of the GPR model's predictive error by the posterior standard deviation in leave-one-out cross-validation. A multiplicative update scheme is proposed for solving the maximum likelihood estimation problem under non-negative constraints. While we provide a proof of monotonic convergence for certain special cases, the multiplicative scheme has empirically demonstrated monotonic convergence behavior in virtually all our numerical experiments. We show that the presented method can pinpoint corrupted samples and lead to better regression models when trained on synthetic and real-world scientific data sets.

Keywords: supervised learning, regression, uncertainty quantification, label noise, kernel method, Gaussian process regression, regularization, heteroscedasticity, maximum likelihood estimation, multiplicative update, graph, computational chemistry

1 Introduction

Machine learning algorithms that can generate robust models from noisy data are of great practical importance. Here, the noisy labels are considered tempered from their unobserved "clean" versions either by an adversary or due to an error in the data acquisition process. Two intertwining goals are usually involved when dealing with noisy labels, *i.e.* 1) to achieve model and training robustness in the presence of noise and 2) to identify and correct noisy sample points while providing feedback to the data collection mechanism.

Depending on the type of labels that are of interest, the focus of existing work is often split between discrete labels and continuous labels. A major body of research has been carried out for the former case regarding binary and categorial labels [1, 2] with a plethora of theoretical work concerning topics such as lower bounds on the sample complexity [3], sample-efficient strategies for learning binary-valued functions [4], empirical risk minimization [5], learnability of linear threshold functions [6], and the role of loss functions [7, 8]. A particular application area of interest is image classification and visual recognition. [9, 10, 11, 12, 13, 14, 15]. Many works targeting both robust training and noisy label identification have been proposed. For example, Wu *et al.* presented an algorithm for recognizing incorrectly labeled samples using an iterative topological filtering process [16]. Tanaka *et al.* proposed a joint optimization framework for learning deep neural network parameters and estimating true labels for image classification problems by an alternating update of network parameters and labels [17]. The strategy of sample selection and using the small-loss trick have also given rise to several algorithms and frameworks for identifying and correcting noisy classification labels [18, 19, 20, 21, 22].

In this paper, our aim is at regression using *a mixture of clean and noisy real-valued labels*. One motivation for this is to screen high-throughput computer simulation data sets, where the results may be corrupted due to random faults but are otherwise of high precision. There are comparably fewer pieces of work on this topic, despite that continuous-label data is ubiquitous and essential in the context of machine learning for scientific modeling. The most relevant previous work concerns heteroscedastic Gaussian process regression. Goldberg *et al.* treated the variance of the noises as a latent function dependent on the input and modeled it with a second Gaussian process [23]. Le *et al.* presented an algorithm to estimate the variance of the Gaussian process locally

using maximum a posterior estimation solved by Newton's method [24]. However, both work regard noise as a property of the underlying Gaussian random field rather than that of individual samples. As a consequence, it is not straightforward to single out anomalous labels on a per-sample basis using their methods. A more general but less relevant paper proposes a general treatment on online learning of real-valued linear and kernel-based predictors using *multiple copies* of each example [25]. To the best of our knowledge, simultaneous robust training and sample-wise noise identification for real-valued regression remains an open problem.

This paper proposes a new algorithm that can identify noise labels while generating accurate GPR models using data sets of high noise rates. The method essentially detect noisy labels using a per-sample heteroscedastic noise model, which leads to a Tikhonov regularization term with independent values for each sample. While various forms of regularization have been widely used in GPR to control overfitting, our work is the first to relate heteroscedastic regularization with noisy label identification. Optimizing the noise model using maximum likelihood estimation results in the reconciliation of GPR leave-one-out cross-validation errors and uncertainties. A simple multiplicative update scheme, which is monotonically converging for optimizing the noise model, underpins the proposed method's practical value.

2 Preliminaries

Notations Upper case letters, *e.g.* M, denote matrices. Bold lower case letters, *e.g.* $\mathbf{a}$, denote column vectors. Regular lower case letters, *e.g.* x, denote scalars. Vectors are assumed to be column vectors by default. $\mathrm{diag}\,(\mathbf{a})$ denotes a diagonal matrix whose diagonal elements are specified by $\mathbf{a}$. $\mathbf{diag}\,(A)$ denotes a vector formed by the diagonal elements of A. $\mathbf{1}^i$ denotes the i-th canonical base vector, *i.e.* an 'indicator' vector whose i-th element is 1 and all other elements are 0. $\odot$ is the Hadamard product, *i.e.* the matrix elementwise product operation.

Gaussian process regression Given a dataset $\mathcal{D}$ of N samples points $\{(\mathbf{x}_i, y_i); \mathbf{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}, i = 1, \ldots, N\}$ and a covariance function, interchangeably called a kernel, $\kappa : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$, a Gaussian process regression (GPR) model [26] can be learned by treating the labels as instantiations of random variables from a Gaussian random field.

For an unknown sample at location $\mathbf{x}_*$, the prediction for its label y_* made by the GPR model takes the form of a posterior normal distribution with mean $\mu_* = \mathbf{k}_* K^{-1} \mathbf{y}$ and variance $\sigma_*^2 = \kappa(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^\mathsf{T} K^{-1} \mathbf{k}_*$. Here, $K_{ij} \doteq \kappa(\mathbf{x}_i, \mathbf{x}_j)$ is the pairwise matrix of prior covariance between the training samples as computed by the covariance function; $(\mathbf{k}_*)_i \doteq \kappa(\mathbf{x}_*, \mathbf{x}_i)$ is the vector of prior covariance between $\mathbf{x}_*$ and the training samples; $\mathbf{y} \doteq [y_1, \ldots, y_N]^\mathsf{T}$ is a vector containing all training labels. Note that we have made the assumption, without loss of generality, that the prior means of the labels are all zero.

The likelihood of data given a GPR model, which describes how well the model can explain and fit the observed data, usually takes the form of a multivariate normal:

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = (2\pi)^{N/2} |K|^{-1/2} \exp\left[-\frac{1}{2}\mathbf{y}^\mathsf{T} K^{-1} \mathbf{y}\right], \qquad (1)$$

where $\boldsymbol{\theta}$ is a list of kernel hyperparameters. Training a GPR model often involves maximum likelihood estimation of $\boldsymbol{\theta}$ which attemps to find $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\mathrm{argmax}}\, p(\mathcal{D} \mid \boldsymbol{\theta})$.

3 Noisy Label Detection

3.1 A Motivating Example

Consider the toy problem where a single sample point $(\mathbf{x}_k, y_k)$ from a dataset $\mathcal{D}$ might contain label noise. To figure out to what degree we could trust y_k in absence of knowledge about the ground truth, an intuition is to examine the posterior likelihood of the suspicious point given a GPR model trained only on the clean labels. Figure 1 illustrates this idea using three versions of a data set with twelve clean labels and one potentially noisy label. Using the posterior distribution of the GPR predictions, we can infer the trustworthiness of the label by examining whether an estimate for the magnitude of the noise need to be added to the prior covariance matrix K to bound the label by the predicted 1σ confidence interval.

The example above is indeed closely related to leave-one-out cross-validation (LOOCV) using GPR. The difference between y_k and the clean-sample GPR prediction on x_k is formally the leave-one-out cross-validation error at sample k by a GPR model trained on the whole dataset, which can be expressed in closed form as $e_k = \frac{(K^{-1}\mathbf{y})_k}{(K^{-1})_{kk}}$. The width of the 1σ confidence interval is the GPR leave-one-out posterior standard deviation at x_k, which can also be expressed in closed form as $s_k = \sqrt{\frac{1}{(K^{-1})_{kk}}}$. Our judgement regarding whether y_k is noisy is made by comparing the magnitude of e_k and s_k: the label is regarded as clean if the magnitude of prior noise added to ensure $e_k < s_k$ is small.

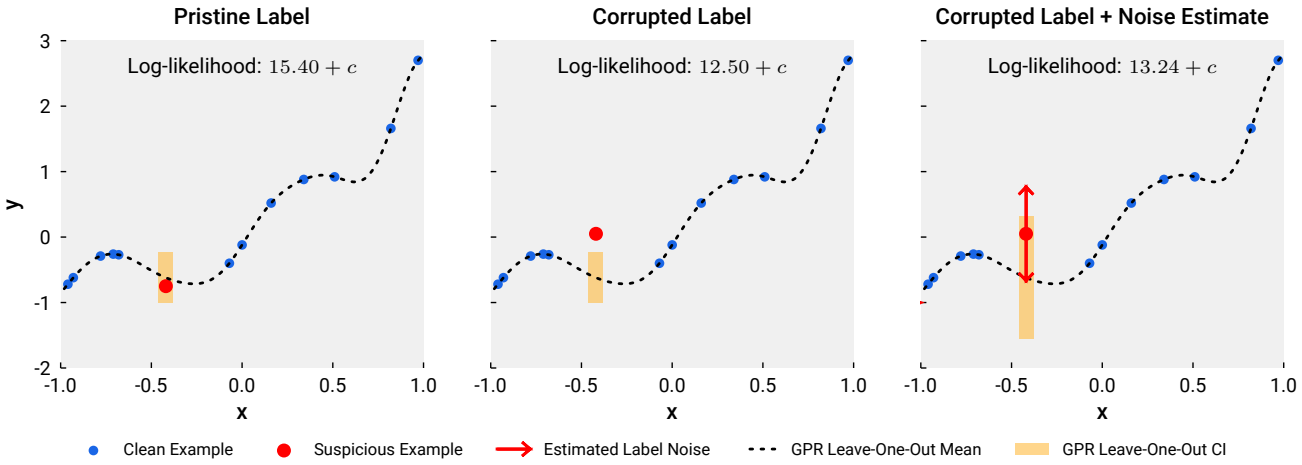

Figure 1: Given a dataset of many clean labels and one potentially noisy label, we could use a GPR model trained on only the clean samples, *i.e.* a leave-one-out model, to infer the noisiness of the suspicious label. In the left panel, the label of question is indeed clean and is bound by the 1σ confidence interval of the leave-one-out model. In the middle panel, the label does contain noise and hence breaks away from the leave-one-out confidence interval. In the right panel, we attempt to bound the noisy label again by the confidence interval by adding an estimate of the noise magnitude into the prior covariance of the sample point. Such estimate results in a higher likelihood of the data and thus confirms that the label contains noise.

3.2 Leave-One-Out Cross-Validation and Heteroscedastic Tikhonov Regularization

Now we formalize the algorithm for noisy label detection using leave-one-out cross-validation with GPR. Note that a difficulty for directly applying the decision process as depicted in Figure 1 to a data set where many samples are noisy is that it is unclear which samples can be trusted to bootstrap the cross validation process. Instead, the identification for the noisy labels is learned via an optimization process as detailed below.

We assume that each observed label $y_i = f(\mathbf{x}_i) + \varepsilon_i$ is a combination of the ground truth label $f(\mathbf{x}_i)$ with an additive Gaussian noise $\varepsilon_i \sim \mathcal{N}(0, \sigma_i)$ of zero mean and sample-specific variance σ_i. Further assuming that the noise variables are mutually independent and also independent of the true labels, *i.e.* $\mathrm{Cov}\left[\varepsilon_i, \varepsilon_j\right] = \delta_{ij}\sigma_i\sigma_j$ and $\mathrm{Cov}\left[y_i, \varepsilon_j\right] \equiv 0\ \forall i, j$, the GPR covariance matrix of this training set then becomes

$$\mathcal{K} = K + \Sigma, \tag{2}$$

where K is the prior covariance matrix as introduced in Section 2, $\Sigma = \mathrm{diag}\,(\boldsymbol{\sigma}) = \mathrm{diag}\left([\sigma_1, \ldots, \sigma_N]^\mathsf{T}\right)$ is the diagonal covariance matrix between the noise terms. Note that Σ can be regarded as a heteroscedastic generalization of the basic Tikhonov regularization term where $\Sigma = \sigma\mathbb{I}$ is uniform among all samples.

To determine Σ, which quantifies the error each label contains, we seek to maximize the likelihood of the dataset given the regularized kernel matrix $\mathcal{K}$. By plugging $\mathcal{K}$ into Equation (1) and assuming that $\mu = 0$, we obtain the following negative log-likelihood function after dropping constant multiplicative factors:

$$\mathcal{L}(\mathbf{y} \mid \mathbf{X}, \kappa, \Sigma) \doteq -\log p(\mathbf{y} \mid \mathbf{X}, \kappa, \Sigma) \propto \log|\mathcal{K}| + \mathbf{y}^\mathsf{T}\mathcal{K}^{-1}\mathbf{y} + c. \tag{3}$$

This leads to the following constrained optimization problem:

$$\begin{aligned} &\underset{\sigma,\theta}{\operatorname{argmin}}\ \mathcal{L}(\mathbf{y} \mid \mathbf{X}, \kappa, \Sigma) \\ &\text{subject to } \sigma_i \geq 0\ \forall\ i. \end{aligned} \tag{4}$$

Following the derivation in Appendix A, the gradient of (3) with respect to $\boldsymbol{\sigma}$ is

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{\sigma}} = \mathrm{diag}\left(\mathcal{K}^{-1}\right) - \left(\mathcal{K}^{-1}\mathbf{y}\right) \odot \left(\mathcal{K}^{-1}\mathbf{y}\right). \tag{5}$$

Thus, a necessary condition for $\mathcal{L}$ to reach an optimum is

$$\mathcal{K}^{-1}{}_{ii} - \left(\mathcal{K}^{-1}\mathbf{y}\right)_i^2 = 0\ \forall\ i. \tag{6}$$

Since $\mathcal{K}^{-1}{}_{ii} > 0$ due to the fact that $\mathcal{K}$ and κ are positive definite, condition (6) can be rearranged as

$$\left[\frac{\left(\mathcal{K}^{-1}\mathbf{y}\right)_i}{\mathcal{K}^{-1}{}_{ii}}\right]^2 = \frac{1}{\mathcal{K}^{-1}{}_{ii}}\ \forall\ i \tag{7}$$

Recall that $\frac{(\mathcal{K}^{-1}\mathbf{y})_i}{\mathcal{K}^{-1}{}_{ii}}$ and $\frac{1}{\mathcal{K}^{-1}{}_{ii}}$ are the leave-one-out cross-validation error and posterior variance of the regularized GPR model for sample i, respectively. What eq. (7) conveys is that

> LOOCV error = LOOCV confidence interval $\forall\ i$
> when $\boldsymbol{\sigma}$ optimizes $\mathcal{L}$.

In other words, the maximum likelihood estimation of the noise terms seeks to bound the the leave-one-out cross validation errors on the training set by the predictive uncertainty, thus minimizing the surprise incurred by any inconsistency between the labels and the prior covariance matrix.

In Figure 2, we compare the proposed formulation against the basic Tikhonov regularization using three synthetic examples. In the first example, the ground truth function $f(x) = \cos(3\pi x) + \sin(\pi x) + 2x^2$ are measured at 24 points drawn from a uniform grid between $[-1, 1]$ perturbed with i.i.d. noises $\sim \mathcal{N}(0, 0.05)$, while 10 of the 24 labels are contaminated with i.i.d. noises $\sim \mathcal{N}(0, 0.75)$. In the second example, we use the setup in Ref. [23] with a reduced sample count of 30 while contaminating 20 of the samples. In the third example, we use the setup in Ref. [24] with a reduced sample count of 50 while contaminating 33 of the samples. The noise parameters in both methods are optimized by multiple maximum likelihood estimation runs from randomized initial guesses. The proposed method consistently delivers better performance in terms of identifying noisy labels and learning accurate regression models.

4 Solution Techniques

4.1 A Multiplicative Update Scheme

It is challenging to derive closed-form solutions for problem (4) due to the non-convex and nonlinear nature of the loss function. While solving problem (4) using a gradient-based optimizer might be convenient, especially for the joint optimization of $\boldsymbol{\sigma}$ and $\boldsymbol{\theta}$ by concatenating $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\sigma}}$ (5) with $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} = \operatorname{tr}\left(\boldsymbol{\mathcal{K}}^{-1} \frac{\partial K}{\partial \boldsymbol{\theta}}\right) - \left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right)^{\mathsf{T}} \frac{\partial K}{\partial \boldsymbol{\theta}} \left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right)$, the rate of convergence could be slow as demonstrated in Figure 3. Meanwhile, The problem size, which scales linearly with the number of samples, practically prevents the usage of more sophisticated algorithms such as L-BFGS-B.

Alternatively, observe that the two terms in the right hand side of (5), *i.e.* $\operatorname{diag}\left(\boldsymbol{\mathcal{K}}^{-1}\right)$ and $\left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right) \odot \left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right)$, both contain only positive elements due to the positive-definiteness of $\boldsymbol{\mathcal{K}}$ and the Hadamard product, respectively. Thus, the direction of the optimization procedure, as indicated by the signs of the gradient elements, is completely determined by the relative magnitude of the elements of the two terms. Intuitively, if ${\boldsymbol{\mathcal{K}}^{-1}}_{ii}$ is smaller than $\left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right)_i^2$ for a certain i, then the likelihood function will have a negative slope along σ_i, prompting us to increase σ_i in order to make $\mathcal{L}$ decrease.

This observation inspires the following multiplicative update scheme for optimizing $\mathcal{L}$:

$$\sigma_i^{(t+1)} = \sigma_i^{(t)} \cdot \frac{\left(\boldsymbol{\mathcal{K}}^{(t)^{-1}}\mathbf{y}\right)_i^2}{\operatorname{diag}\left(\boldsymbol{\mathcal{K}}^{(t)^{-1}}\right)_i}, \tag{8}$$

where the superscript $^{(t)}$ indicates the version of $\boldsymbol{\sigma}$ at iteration step t. Following the same rationale as in the previous paragraph, when the slope is negative along a certain σ_i, the multiplicative factor will be greater than one, thus increasing the value of σ_i after the update. When the gradient is positive, the multiplicative factor will be smaller than one to decrease σ_i. The non-negativity constraint on $\boldsymbol{\sigma}$ is automatically honored as long as the starting point $\boldsymbol{\sigma}^{(0)}$ is non-negative.

It is easy to verify that the stationary points of the log-likelihood functions are the fixed points of the multiplicative update rule. To see that, note that $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\sigma}} = \mathbf{0}$ implies $\operatorname{diag}\left(\boldsymbol{\mathcal{K}}^{-1}\right)_i = \left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right)_i^2 \ \forall\ i$. In that case, we have all the update coefficients $\frac{\operatorname{diag}\left(\boldsymbol{\mathcal{K}}^{-1}\right)_i}{\left(\boldsymbol{\mathcal{K}}^{-1}\mathbf{y}\right)_i^2} = 1$, which indicates that the recurrence relationship has reached a fixed point. Also note that the zero vector $\mathbf{0}$ is a trivial fixed point of the multiplicative update rule.

Surprisingly, the multiplicative update scheme has exhibited monotonic convergence behavior in virtually all synthetic and real-world cases that we have tested. The rate of convergence is also very fast as shown in Figure 3, in which we compare the scheme against several gradient descent and quasi-Newton methods.

The fact that the multiplicative update scheme for optimizing $\boldsymbol{\sigma}$ can converge monotonically in a very efficient manner creates many opportunities for the joint optimization of $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$. One possibility is to cast the problem into the bi-level optimization paradigm, where $\boldsymbol{\theta}$ is treated by an upper-level optimizer, while the multiplicative update of $\boldsymbol{\sigma}$ as the lower-level optimizer. Another possibility is to interleave the optimization of $\boldsymbol{\sigma}$ and $\boldsymbol{\theta}$ in alternating steps.

One last note is that the multiplicative update scheme is also empirically observed to be monotonically converging when $\Sigma = \sigma\mathbb{I}$. In that case, the update rule is $\sigma^{(t+1)} = \sigma^{(t)} \cdot \frac{\left(\boldsymbol{\mathcal{K}}^{(t)^{-1}}\mathbf{y}\right)^{\mathsf{T}}\left(\boldsymbol{\mathcal{K}}^{(t)^{-1}}\mathbf{y}\right)}{\operatorname{Tr}\left[\boldsymbol{\mathcal{K}}^{(t)^{-1}}\right]}$.

4.2 Convergence analysis

In this section, we attempt to provide some theoretical insight into the observed monotonic convergence behavior of the multiplicative update scheme. However, a rigorous proof for the general case is not available yet.

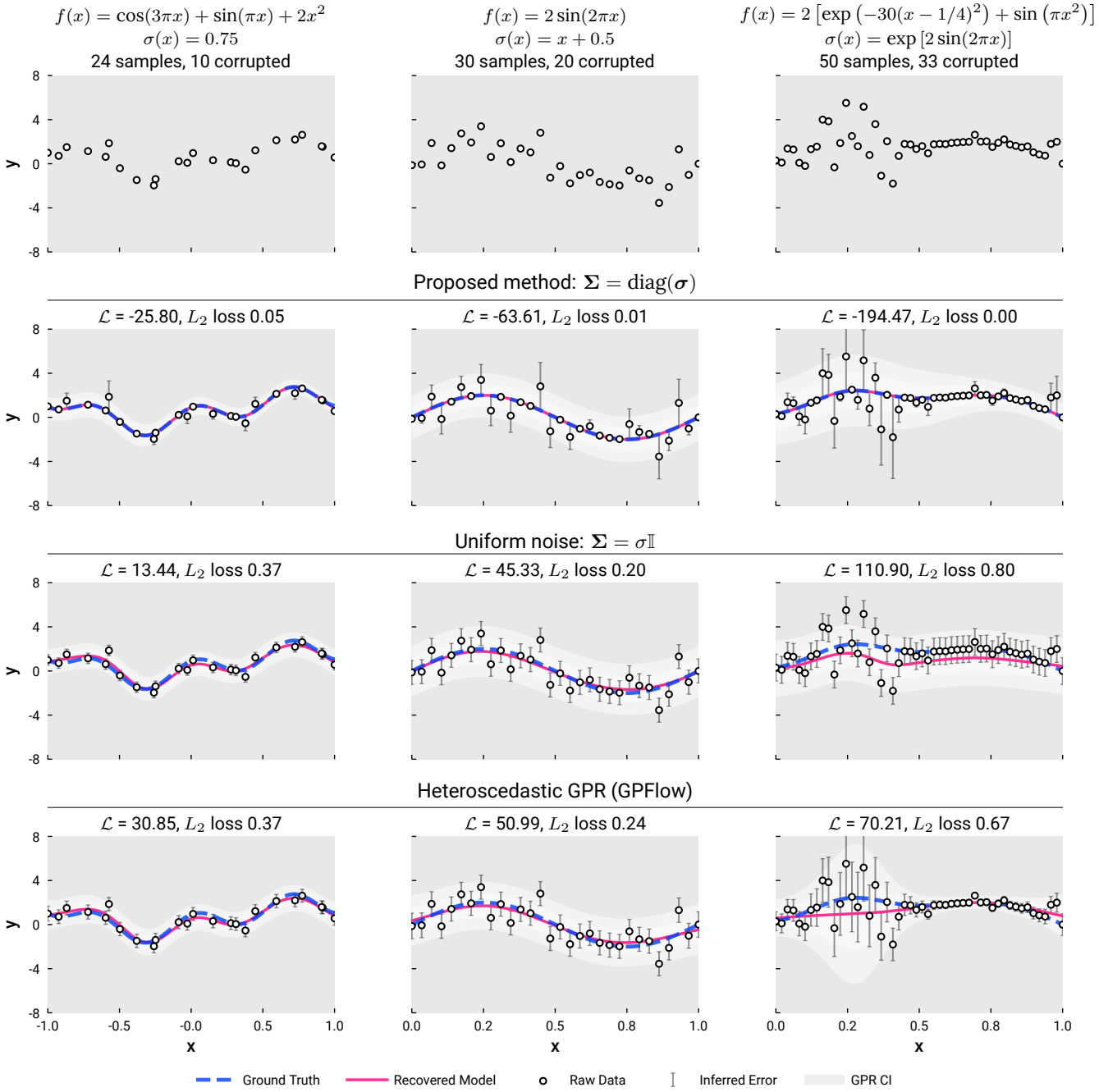

Figure 2: A comparison between the algorithm proposed in Section 3.2, the uniform noise model, and the heteroscedastic GPR implementation of GPFlow [27]. Three sets of 1D functions and noise distributions are tested for illustrative purposes.

Theorem 1. *For a special case where* $K = \mathrm{diag}(\mathbf{k})$ *is a diagonal matrix, if the optimization problem* (4) *has solutions, then the solution is unique and the multiplicative update scheme* (8) *monotonically converges to the unique solution.*

Proof. We use the fixed-point theorem to get the result. When K is diagonal, $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\sigma}} = \mathbf{0}$ can be solved independently for each i which yields the exact result:

$$\frac{y_i^2}{(K_{ii} + \sigma_i^*)^2} = \frac{1}{(K_{ii} + \sigma_i^*)^2} \qquad \Rightarrow \qquad \sigma_i^* = y_i^2 - K_{ii}.$$

Since $K = \mathrm{diag}(\mathbf{k})$ is a symmetric, positive definite matrix, the constraint $\sigma_i^* \geq 0$ implies $0 < K_{ii}/y_i^2 \leq 1$. For such

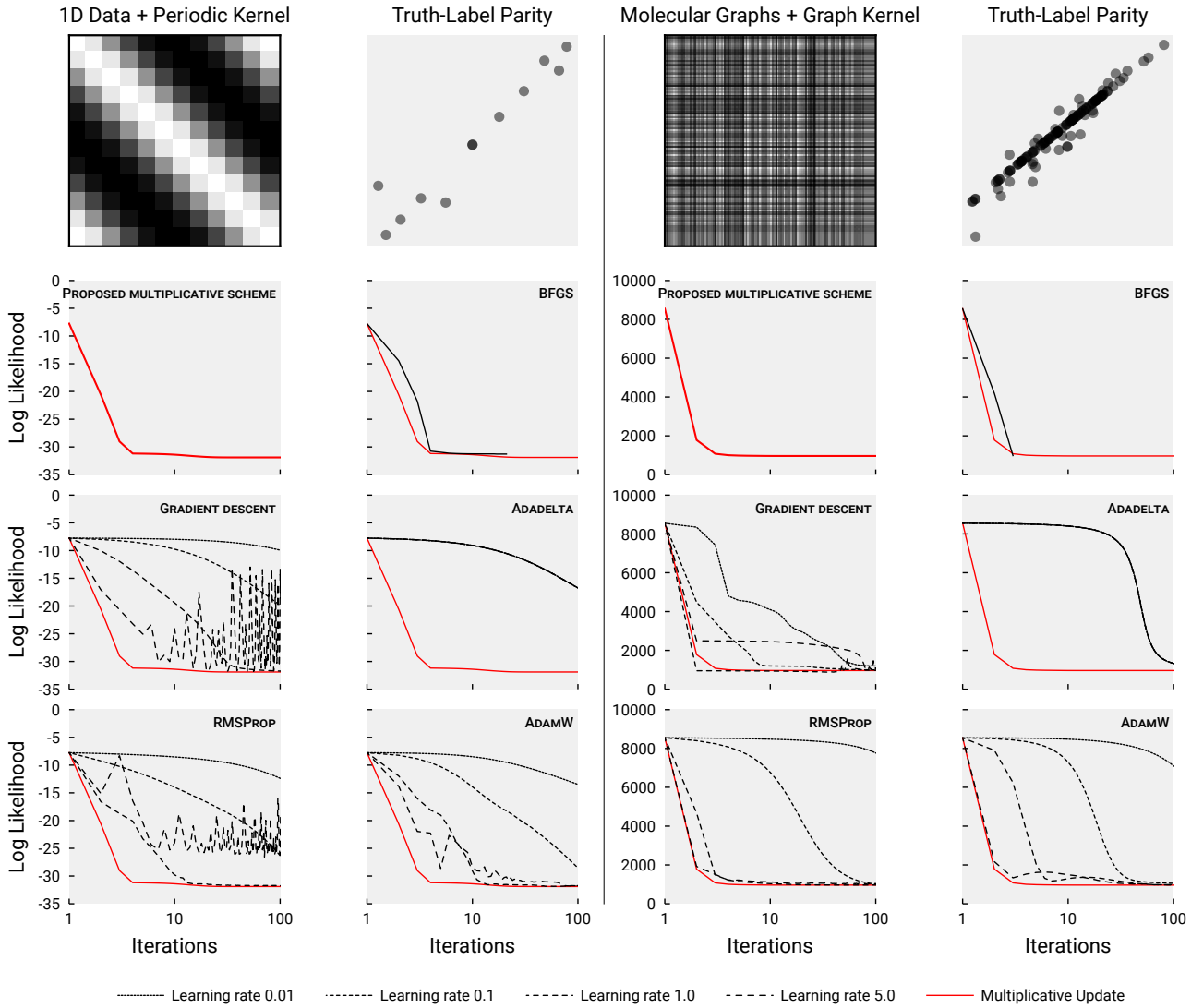

Figure 3: An empirical comparison of the multiplicative update scheme as in eq. (8) against the quasi-Newton method L-BFGS-B and four variants of gradient descent algorithms. The multiplicative update scheme outperforms virtually all other algorithms for optimizing σ in terms of the rate of convergence and the number of function evaluations.

case, iteration scheme (8) reduces to

$$\sigma_i^{(t+1)} = \sigma_i^{(t)} \frac{y_i^2}{\sigma_i^{(t)} + K_{ii}}, \qquad 1 \le i \le N,$$

which can be reformulated as

$$\frac{1}{\sigma_i^{(t+1)}} = \frac{1}{y_i^2} + \frac{K_{ii}}{y_i^2} \frac{1}{\sigma_i^{(t)}} = g_i\left(\frac{1}{\sigma_i^{(t)}}\right).$$

Set $p_i^{(t+1)} = 1/\sigma_i^{(t+1)}$ and set $p_i^* = 1/\sigma_i^*$ to be the exact fixed point. Now, the mean-value theorem implies the $(t+1)$-step error can be bounded as

$$\begin{aligned} \left|p_i^{(t+1)} - p_i^*\right| &= \left|g\left(p_i^{(t)}\right) - g\left(p_i^*\right)\right| \\ &= \left|g'\left(\xi^{(t+1)}\right)\right| \left|p_i^{(t)} - p_i^*\right| \\ &= \frac{K_{ii}}{y_i^2} \left|p_i^{(t)} - p_i^*\right|. \end{aligned}$$

Given the prerequisite $0 < K_{ii}/y_i^2 \le 1$, when $K_{ii}/y_i^2 = k_i < 1$, we can get the monotonic convergence result:

$$\left|p_i^{(t+1)} - p_i^*\right| \le k_i^n \left|p_i^{(0)} - p_i^*\right| \to 0 \qquad \text{as} \qquad n \to \infty. \tag{9}$$

If otherwise $K_{ii}/y_i^2 = k_i = 1$, we have

$$p_i^{(t+1)} = \frac{n}{y_i^2} + p_i^{(0)} \to +\infty \qquad \text{as} \qquad n \to \infty, \tag{10}$$

which corresponds to a monotonically convergent $\sigma_i^{(t)} \to 0$ since $\sigma_i^{(t)} = 1/p_i^{(t)}$. Hence, for this special case, we conclude that if there exists a solution to (4), then the solution is unique and the multiplicative update scheme (8) will monotonically converge to this unique solution. □

Immediately, we can get the convergence of the loss function as below.

Corollary 1.1. *For the special case where $K = \mathrm{diag}(\mathbf{k})$ is a diagonal matrix, if the optimization problem* (4) *has solutions, then the multiplicative update scheme* (8) *generates a convergent sequence $\mathcal{L}(\boldsymbol{\sigma}^{(t)}) \to \mathcal{L}(\boldsymbol{\sigma}^*)$, where $\mathcal{L}(\boldsymbol{\sigma}^*)$ is the optimal loss. If the exact solution $\sigma_i^* > 0$ for all $1 \le i \le N$, then the convergence rate is R-linear, otherwise, it is R-superlinear.*

Proof. When $K = \mathrm{diag}(\mathbf{k})$ is a diagonal matrix, using the definition of $\mathcal{L}$ (3), we can get the loss at the t-th iteration:

$$\mathcal{L}(\boldsymbol{\sigma}^{(t)}) \propto \sum_{i=1}^{N} \log\left(\sigma_i^{(t)} + K_{ii}\right) + \sum_{i=1}^{N} \frac{y_i^2}{\sigma_i^{(t)} + K_{ii}} \tag{11}$$

To get the R-convergence of $\mathcal{L}$, we only need to prove that the upper bound of $\left|\mathcal{L}(\boldsymbol{\sigma}^{(t)}) - \mathcal{L}(\boldsymbol{\sigma}^*)\right|$ converges to 0. Consider the i-th term in the first summation in (11), we have the error estimate:

$$\left|\log\left(\sigma_i^{(t)} + K_{ii}\right) - \log\left(\sigma_i^* + K_{ii}\right)\right| \le \frac{1}{K_{ii}} \left|\sigma_i^{(t)} - \sigma_i^*\right|. \tag{12}$$

Here we used the Lipschitz condition of $\log(x)$ on domain $[K_{ii}, +\infty)$, i.e. $\left|\log(x) - \log(y)\right| \le \left|x - y\right| / K_{ii}$. For the i-th term in the second summation in (11), we also have

$$\left|\frac{y_i^2}{\sigma_i^{(t)} + K_{ii}} - \frac{y_i^2}{\sigma_i^* + K_{ii}}\right| \tag{13}$$

$$\le \left|\frac{y_i^2}{\left(\sigma_i^{(t)} + K_{ii}\right)\left(\sigma_i^* + K_{ii}\right)}\right| \left|\sigma_i^{(t)} - \sigma_i^*\right| \tag{14}$$

$$\le \frac{y_i^2}{K_{ii}\left(\sigma_i^* + K_{ii}\right)} \left|\sigma_i^{(t)} - \sigma_i^*\right| \tag{15}$$

Combining estimate (12)-(13), we can find a constant C such that

$$\left|\mathcal{L}(\boldsymbol{\sigma}^{(t)}) - \mathcal{L}(\boldsymbol{\sigma}^*)\right| \le C \left\|\boldsymbol{\sigma}^{(t)} - \boldsymbol{\sigma}^*\right\|_\infty. \tag{16}$$

The above upper bound converges to 0 linearly if the exact solution $\sigma_i^* > 0$ for all $1 \le i \le N$. Otherwise, we can only get the superlinear convergence according to (10) because the convergence rate is of the order $O(1/n)$ in the j-th direction where $\sigma_j^* = 0$. □

4.3 Penalty and Sparsification

If a sparse solution is desired or if we want to prevent the method from being too aggressive in calling out noisy labels, an ℓ_p penalty term can be introduced into the loss function:

$$\mathcal{L}_\lambda^p = \mathcal{L} + \lambda \|\boldsymbol{\sigma}\|_p^p. \tag{17}$$

Due to the non-negative nature of the gradient of the ℓ_p term, it can also be incorporated into the multiplicative update scheme as

$$\sigma_i^{(t+1)} = \sigma_i^{(t)} \cdot \frac{\left(\boldsymbol{\mathcal{K}}^{(t)^{-1}}\mathbf{y}\right)_i^2}{\mathrm{diag}\left(\boldsymbol{\mathcal{K}}^{(t)^{-1}}\right)_i + \lambda\, p \left|\sigma_i^{(t)}\right|^{p-1}}. \tag{18}$$

5 Experiments

We demonstrate the capability of the proposed method using the QM7 [28, 29] and QM9 [30] data sets of small organic molecules. To carry out the experiments, we add normally distributed noise to the labels in the data sets with a series of combinations of noise rates and noise levels. Here, noise rate is defined as the percentage of labels that we will artificially corrupt, while noise level is the ratio between the noise and the standard deviation of the pristine labels.

The data sets consists of minimum-energy 3D geometry of small organic molecules and their associated properties. The covariance function κ is a marginalized graph kernel defined between molecular graphs that encode the spatial arrangement and topology of the molecules [31].

From Table 1, we can see that our proposed method can consistently improve the accuracy of the trained GPR model even in the presence of very high noise rate. Moreover, our method can also capture a high fraction of the noisy labels in most scenarios. Generally speaking, the ability of the method to distinguish noisy and clean labels generally increases with noise level but decreases with noise rate. This indicates that large numbers of small perturbations within the model's confidence interval is likely to cause the most degradation to the performance of the trained model.

6 Conclusion

A method that uses Gaussian process regression to identify noisy real-valued labels is introduced to improve models trained on data sets of high output noise rate. To infer the magnitude of noise on a per-sample basis, we use maximum likelihood estimation to optimize a heteroscedastic noise model and learn a sample-wise Tikhonov regularization term. We show that this is

Table 1: Label Noise - Rate/Level: the percentage of corrupted labels and the ratio between the noise and the standard deviation of the pristine labels; R^2: the coefficient of determination between the inferred and actual label noise; AUC: area under the ROC curve of a 'noisy label' classifier that thresholds the learned σ_i; Precision at Recall Level: precision of the classifier at specified recall levels. Regression accuracy - plain/basic/full: $\Sigma = 0,\ \sigma\mathbb{I},\ \mathrm{diag}(\sigma)$, respectively.

QM7 Atomization Energy

Label Noise		Noisy Label Detection (Thresholding on σ_i)				GPR Accuracy (5-fold CV MAE (kcal/mol))		
Rate	Level	R^2	AUC	Precision at Recall Level		(Pristine Data: 1.05)		
				70%	95%	Plain	Basic	Full
10%	10%	0.59	.952	76.8%	27.9%	3.75	2.26	1.14
	50%	0.98	.991	96.6%	79.7%	17.78	4.93	1.12
	100%	1.00	.996	97.3%	83.7%	34.59	7.05	1.12
	200%	1.00	.997	98.6%	96.9%	68.83	10.21	1.12
30%	10%	0.86	.949	92.4%	57.8%	6.55	2.85	1.20
	50%	0.99	.988	99.3%	92.3%	31.89	6.68	1.24
	100%	1.00	.994	99.3%	95.3%	64.18	9.83	1.23
	200%	1.00	.997	99.5%	98.9%	128.08	15.76	1.25
50%	10%	0.92	.944	95.7%	77.7%	8.32	3.19	1.34
	50%	1.00	.987	99.6%	96.5%	42.86	7.13	1.38
	100%	1.00	.993	99.7%	97.6%	82.79	12.49	1.40
	200%	1.00	.995	99.8%	99.5%	167.64	16.23	1.40
70%	10%	0.89	.912	96.7%	81.1%	10.10	3.57	1.90
	50%	1.00	.984	99.8%	98.0%	49.01	8.12	1.78
	100%	1.00	.992	99.9%	98.9%	99.44	12.00	1.70
	200%	1.00	.996	99.9%	99.7%	198.57	17.31	1.72
90%	10%	0.81	.818	97.3%	92.0%	11.74	3.76	3.94
	50%	0.99	.961	99.9%	98.1%	57.50	8.59	4.07
	100%	1.00	.982	99.9%	99.3%	112.77	13.20	3.10
	200%	1.00	.992	100.0%	99.9%	221.00	19.08	2.90

QM9 8K Subset Atomization Energy

Label Noise		Noisy Label Detection (Thresholding on σ_i)				GPR Accuracy (5-fold CV MAE (kcal/mol))		
Rate	Level	R^2	AUC	Precision at Recall Level		(Pristine Data: 1.45)		
				70%	95%	Plain	Basic	Full
10%	10%	0.69	.952	72.4%	32.5%	5.61	2.98	1.54
	50%	0.99	.983	91.0%	72.9%	26.02	5.80	1.58
	100%	1.00	.993	94.1%	84.3%	52.52	7.99	1.55
	200%	1.00	.994	96.7%	89.7%	104.55	10.74	1.55
30%	10%	0.86	.945	90.2%	60.5%	8.99	3.63	1.71
	50%	0.99	.986	97.5%	90.9%	46.06	7.48	1.72
	100%	1.00	.989	98.1%	93.5%	89.46	11.03	1.72
	200%	1.00	.995	99.0%	97.3%	181.18	15.78	1.70
50%	10%	0.91	.937	94.8%	75.4%	12.09	4.14	1.91
	50%	0.99	.982	98.9%	95.3%	55.84	7.73	1.94
	100%	1.00	.988	99.1%	96.7%	117.07	10.17	1.95
	200%	1.00	.995	99.7%	99.1%	239.49	16.50	1.91
70%	10%	0.89	.897	96.0%	79.0%	14.19	4.34	2.59
	50%	0.99	.979	99.5%	97.4%	71.74	8.88	2.39
	100%	1.00	.987	99.6%	98.4%	137.61	12.37	2.28
	200%	1.00	.994	99.8%	99.5%	281.35	19.02	2.31
90%	10%	0.80	.809	97.2%	91.3%	16.01	4.87	5.07
	50%	0.99	.955	99.9%	97.7%	80.58	9.39	5.18
	100%	1.00	.971	99.8%	98.5%	157.74	14.64	4.99
	200%	1.00	.990	99.9%	99.8%	317.14	18.79	3.85

QM9 8K Subset Polarizability

Label Noise		Noisy Label Detection (Thresholding on σ_i)				GPR Accuracy (5-fold CV MAE (Å^3))		
Rate	Level	R^2	AUC	Precision at Recall Level		(Pristine Data: 0.51)		
				70%	95%	Plain	Basic	Full
10%	10%	-12.17	.652	16.5%	10.6%	0.52	0.62	0.56
	50%	0.45	.864	53.4%	12.0%	0.93	0.81	0.56
	100%	0.83	.912	69.6%	18.5%	1.49	1.00	0.57
	200%	0.97	.961	82.0%	40.1%	3.04	1.28	0.57
30%	10%	-4.62	.635	39.7%	31.4%	0.58	0.67	0.63
	50%	0.81	.852	73.3%	36.9%	1.45	0.96	0.60
	100%	0.94	.920	92.1%	53.9%	2.69	1.19	0.61
	200%	0.98	.939	91.6%	57.5%	5.41	1.45	0.63
50%	10%	-2.15	.645	60.6%	52.5%	0.67	0.69	0.63
	50%	0.83	.844	82.6%	57.1%	1.73	1.03	0.65
	100%	0.96	.916	95.1%	68.7%	3.36	1.28	0.74
	200%	0.99	.942	96.1%	76.9%	7.21	1.60	0.71
70%	10%	-1.31	.630	76.4%	70.7%	0.67	0.72	0.65
	50%	0.83	.816	89.7%	74.8%	2.25	1.15	0.82
	100%	0.95	.892	97.1%	81.2%	4.07	1.37	0.86
	200%	0.98	.919	97.0%	85.2%	8.01	1.75	0.90
90%	10%	-3.23	.606	92.4%	90.7%	0.71	0.74	0.76
	50%	0.69	.778	95.9%	91.5%	2.39	1.13	1.24
	100%	0.87	.863	98.8%	92.2%	4.80	1.41	1.50
	200%	0.96	.891	98.7%	94.5%	9.19	2.00	1.48

QM9 8K Subset Band Gap

Label Noise		Noisy Label Detection (Thresholding on σ_i)				GPR Accuracy (5-fold CV MAE (eV))		
Rate	Level	R^2	AUC	Precision at Recall Level		(Pristine Data: 0.30)		
				70%	95%	Plain	Basic	Full
10%	10%	-61.36	.532	10.6%	10.1%	0.30	0.30	0.28
	50%	-1.85	.728	18.5%	10.7%	0.32	0.32	0.28
	100%	0.30	.822	35.2%	11.9%	0.36	0.34	0.29
	200%	0.82	.883	58.3%	13.2%	0.48	0.40	0.29
30%	10%	-24.46	.531	31.1%	30.3%	0.30	0.30	0.28
	50%	-0.28	.706	42.7%	31.7%	0.34	0.34	0.30
	100%	0.65	.809	65.3%	33.2%	0.44	0.38	0.30
	200%	0.92	.877	83.0%	36.9%	0.74	0.46	0.31
50%	10%	-17.56	.530	51.0%	50.2%	0.30	0.30	0.28
	50%	-0.04	.692	61.4%	51.6%	0.37	0.35	0.31
	100%	0.68	.790	77.4%	53.4%	0.51	0.41	0.33
	200%	0.92	.865	89.9%	57.4%	0.89	0.49	0.34
70%	10%	-15.13	.528	70.9%	69.9%	0.30	0.30	0.28
	50%	-0.10	.675	78.2%	71.0%	0.39	0.36	0.34
	100%	0.60	.753	85.6%	72.3%	0.59	0.43	0.39
	200%	0.88	.823	93.3%	74.5%	1.06	0.53	0.42
90%	10%	-15.16	.517	90.4%	90.0%	0.30	0.30	0.29
	50%	-0.35	.648	92.5%	90.3%	0.41	0.37	0.37
	100%	0.41	.721	94.9%	90.8%	0.63	0.44	0.47
	200%	0.78	.773	97.5%	91.3%	1.16	0.53	0.59

closely related to denoising using leave-one-out cross-validation. A simple multiplicative updating scheme is designed to solve the numerical optimization problem. The scheme monotonically converges in a broad range of test cases and has defeated our extensive efforts in seeking a counterexample. The capability of the noise detection method is demonstrated on both synthetic and real-world scientific data sets.

While we presented a preliminary analysis of the multiplicative update scheme's convergence behavior for a special case, future work is necessary for a thorough determination of the algorithm's region of convergence. There is also strong practical interest in the adaptation of the method to GPR extensions such as those based on non-Gaussian likelihoods and Nyström [32, 33] or hierarchical low-rank approximations [34, 35, 36], as well as in multi-level optimizers that can take advantage of the fast convergence of the multiplicative algorithm for the joint optimization of both the kernel hyperparameters and the noise model.

7 Acknowledgment

This work was supported by the Luis W. Alvarez Postdoctoral Fellowship at Lawrence Berkeley National Laboratory.

References

[1] K. Crammer and D. D. Lee. Learning via Gaussian Herding. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1*, NIPS'10, pages 451–459, Red Hook, NY, USA, 2010. Curran Associates Inc.

[2] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. *Machine Language*, 91(2):155–187, 2013.

[3] J. A. Aslam and S. E. Decatur. On the sample complexity of noise-tolerant learning. *Information Processing Letters*, 57(4):189–195, 1996.

[4] N. Cesa-Bianchi, E. Dichterman, P. Fischer, E. Shamir, and H. U. Simon. Sample-efficient strategies for learning in the presence of noise. *Journal of the ACM*, 46(5):684–719, 1999.

[5] N. Natarajan, I. S. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. In *NIPS*, volume 26, pages 1196–1204, 2013.

[6] T. Bylander. Learning linear threshold functions in the presence of classification noise. In *Proceedings of the Seventh Annual Conference on Computational Learning Theory*, COLT '94, pages 340–347, New York, NY, USA, 1994. Association for Computing Machinery.

[7] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, Classification, and Risk Bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[8] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey. Normalized Loss Functions for Deep Learning with Noisy Labels. In *International Conference on Machine Learning*, pages 6543–6553. PMLR, 2020.

[9] B. Biggio, B. Nelson, and P. Laskov. Support Vector Machines Under Adversarial Label Noise. In *Asian Conference on Machine Learning*, pages 97–112. PMLR, 2011.

[10] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang. Learning From Massive Noisy Labeled Data for Image Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.

[11] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li. Learning From Noisy Labels With Distillation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1910–1918, 2017.

[12] J. Li, R. Socher, and S. C. H. Hoi. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. *arXiv:2002.07394 [cs]*, 2020.

[13] D. Karimi, H. Dou, S. K. Warfield, and A. Gholipour. Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, 65:101759, 2020.

[14] Q. Yao, H. Yang, B. Han, G. Niu, and J. T.-Y. Kwok. Searching to Exploit Memorization Effect in Learning with Noisy Labels. In *International Conference on Machine Learning*, pages 10789–10798. PMLR, 2020.

[15] L. Jiang, D. Huang, M. Liu, and W. Yang. Beyond Synthetic Noise: Deep Learning on Controlled Noisy Labels. In *International Conference on Machine Learning*, pages 4804–4815. PMLR, 2020.

[16] P. Wu, S. Zheng, M. Goswami, D. Metaxas, and C. Chen. A Topological Filter for Learning with Label Noise. *arXiv:2012.04835 [cs]*, 2020.

[17] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa. Joint Optimization Framework for Learning With Noisy Labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5552–5560, 2018.

[18] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels. *arXiv:1804.06872 [cs, stat]*, 2018.

[19] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei. MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.

[20] H. Song, M. Kim, and J.-G. Lee. SELFIE: Refurbishing Unclean Samples for Robust Deep Learning. In *International Conference on Machine Learning*, pages 5907–5915. PMLR, 2019.

[21] H. Wei, L. Feng, X. Chen, and B. An. Combating Noisy Labels by Agreement: A Joint Training Method with Co-Regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13726–13735, 2020.

[22] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama. How does Disagreement Help Generalization against Label Corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR, 2019.

[23] P. W. Goldberg, C. K. I. Williams, and C. M. Bishop. Regression with input-dependent noise: A gaussian process treatment. In *NIPS*, pages 493–499, 1997.

[24] Q. V. Le, A. J. Smola, and S. Canu. Heteroscedastic Gaussian process regression. In *Proceedings of the 22nd International Conference on Machine Learning - ICML '05*, pages 489–496, Bonn, Germany, 2005. ACM Press.

[25] N. Cesa-Bianchi, S. Shalev-Shwartz, and O. Shamir. Online Learning of Noisy Data. *IEEE Transactions on Information Theory*, 57(12):7907–7931, 2011.

[26] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[27] A. G. De G. Matthews, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman. GPflow: A Gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304, 2017.

[28] L. C. Blum and J.-L. Reymond. 970 Million Druglike Small Molecules for Virtual Screening in the Chemical Universe Database GDB-13. *Journal of the American Chemical Society*, 131(25):8732–8733, 2009.

[29] M. Rupp, A. Tkatchenko, K.-R. Muller, O. A. von Lilienfeld, K.-R. R. Müller, O. Anatole Von Lilienfeld, and O. A. von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical Review Letters*, 108(5):58301, 2012.

[30] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1:140022, 2014.

[31] Y.-H. Tang and W. A. de Jong. Prediction of atomization energy using graph kernel and active learning. *The Journal of Chemical Physics*, 150(4):044107, 2019.

[32] C. Williams and M. Seeger. Using the nyström method to speed up kernel machines. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2001.

[33] M. Li, J. T. Kwok, and B.-L. Lu. Making large-scale nyström approximation possible. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 631–638, Madison, WI, USA, 2010. Omnipress.

[34] Y. Liu, W. Sid-Lakhdar, E. Rebrova, P. Ghysels, and X. S. Li. A parallel hierarchical blocked adaptive cross approximation algorithm. *The International Journal of High Performance Computing Applications*, 34(4):394–408, 2020.

[35] E. Rebrova, G. Chávez, Y. Liu, P. Ghysels, and X. S. Li. A Study of Clustering Techniques and Hierarchical Matrix Formats for Kernel Ridge Regression. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 883–892, 2018.

[36] G. Chávez, Y. Liu, P. Ghysels, X. S. Li, and E. Rebrova. Scalable and Memory-Efficient Kernel Ridge Regression. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 956–965, 2020.

Notations The Einstein summation convention for repeating indices is implied for all variables except for i and j.

A Gradient of the negative log-likelihood function

Starting with the negative log-likelihood of the Gaussian process in Equation (3), we can derive its partial derivative with respect to $\boldsymbol{\Sigma}$:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\Sigma}} = \frac{\partial \log \left|\tilde{\mathbf{K}}\right|}{\partial \boldsymbol{\Sigma}} + \frac{\partial \left(\mathbf{y}^\mathsf{T} \tilde{\mathbf{K}}^{-1} \mathbf{y}\right)}{\partial \boldsymbol{\Sigma}} \tag{A.1}$$

$$= \operatorname{tr}\left(\tilde{\mathbf{K}}^{-1} \frac{\partial \tilde{\mathbf{K}}}{\partial \boldsymbol{\Sigma}}\right) - \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)^\mathsf{T} \frac{\partial \tilde{\mathbf{K}}}{\partial \boldsymbol{\Sigma}} \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right) \tag{A.2}$$

$$= \tilde{\mathbf{K}}^{-1}{}_{kl} \frac{\partial \tilde{\mathbf{K}}_{kl}}{\partial \boldsymbol{\Sigma}_{ij}} - \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_k \frac{\partial \tilde{\mathbf{K}}_{kl}}{\partial \boldsymbol{\Sigma}_{ij}} \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_l \tag{A.3}$$

$$= \tilde{\mathbf{K}}^{-1}{}_{kl} \frac{\partial \boldsymbol{\Sigma}_{kl}}{\partial \boldsymbol{\Sigma}_{ij}} - \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_k \frac{\partial \boldsymbol{\Sigma}}{\partial \boldsymbol{\Sigma}_{ij}} \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_l \tag{A.4}$$

$$= \tilde{\mathbf{K}}^{-1}{}_{kl} \delta_{ki}\delta_{lj} - \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_k \delta_{ki}\delta_{lj} \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_l \tag{A.5}$$

$$= \tilde{\mathbf{K}}^{-1}{}_{ij} - \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_i \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)_j \tag{A.6}$$

$$= \tilde{\mathbf{K}}^{-1} - \left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)\left(\tilde{\mathbf{K}}^{-1}\mathbf{y}\right)^\mathsf{T} \tag{A.7}$$