Chemical machine learning with kernels: The key impact of loss functions

Quang Van Nguyen,[1, a)] Sandip De,[2, 3] Junhong Lin,[1] and Volkan Cevher[1]
[1)] *Laboratory for Information and Inference Systems (LIONS)*
École Polytechnique Fédérale de Lausanne, Switzerland
[2)] *Laboratory of Computational Science and Modelling (COSMO), Institute of Materials,*
École Polytechnique Fédérale de Lausanne, Switzerland
[3)] *National Center for Computational Design and Discovery of Novel Materials (MARVEL)*

Machine learning promises to accelerate materials discovery by allowing computational efficient property predictions from a small number of reference calculations. As a result, the literature spent a considerable effort in designing representations that capture basic physical properties so far. In stark contrast, our work focuses on the less-studied learning formulations in this context in order to exploit inner structures in the prediction errors. In particular, we propose to directly optimize basic loss functions of the prediction error metrics typically used in the literature, such as the mean absolute error or the worst case error. We show that a proper choice of the loss function can directly improve the prediction performance in the desired metric, albeit at the cost of additional computations during training. To support this claim, we describe the statistical learning theoretic foundations and provide numerical evidence with the prediction of atomization energies for a database of small organic molecules.

I. INTRODUCTION

Estimating the ground state energy of molecules and crystals is one of the most fundamental topics in computational quantum mechanics. The traditional approach is to use the density functional theory (DFT)[1,2] which solves Schrödinger's equations with extremely expensive calculations. Recently, there is a great deal of interest in the materials design using machine learning at quantum chemistry level based DFT data. This research vein has been supported with strong preliminary evidence that we can simulate relatively large systems, containing thousands of atoms with accurate prediction performance.

As a result, a considerable effort has gone into building machine learning models for purpose of representing the atomic data. In particular, the existing literature, to out knowledge, mainly focuses on the design of kernels along with the so-called "descriptors" or "fingerprints", e.g., bond lengths, bond angles, etc, to tailor the machine learning procedures to capture subtle differences in atomic environments. The resulting machine learning frameworks often use a kernel ridge regression or neural networks with impressive prediction performance.

In stark contrast, our work emphasizes the learning formulations, i.e., the loss functions, which have received very little attention in the same context. To go beyond the root mean squared error (RMSE) metric, we provide learning theoretic arguments to motivate loss functions to improve predictions in the mean absolute error (MAE) and max absolute error (MaxAE) metrics.

MAE had been cited in the very early forecasting literature as a primary measure of performance for forecasting models[3] and has recently come to our attention due to its robustness. MaxAE, on the other hand, is an upper bound for both RMSE and MAE and reflects the prediction with the highest inaccuracy.

In the sequel, we represent the state of a molecule by a sequence $\{(r_k, z_k)\}_{k=1}^K$, where $r_k \in \mathbb{R}^3$ is the position of k-th nuclei and z_k is its charge. This physical state is translated into a vector-like representation $\mathbf{x} \in \mathbb{R}^n$, which is usually required to be invariant with respect to permutational, rotational, reflectional and translational symmetries[4].

The paper is organized as follows. Section II discusses the statistical learning perspective of ground state energy regression problem, including regularized M-estimators, cross-validation method and kernel trick. Section III gives the mathematical details of basic convex optimization and numerical methods to approximate a solution of our novelly proposed models used in predicting ground state energy. Finally, in Section IV, we provide concrete numerical evidence with an already designed kernel for the prediction of atomization energies for a database of small organic molecules and improve the usual kernel ridge regression (KRR) at the expense of more computation.

Notation. The n-dimensional Euclidean space is denoted by $\mathbb{R}^n$. The transpose and the inverse of a positive definite matrix $\mathbf{K}$ are denoted by $\mathbf{K}^\top$ and $\mathbf{K}^{-1}$, respectively. Some common norms of a vector $\mathbf{x} \in \mathbb{R}^n$ are denoted as follows. We define the ℓ_1-norm as $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$; the ℓ_2-norm as $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^n |x_i|^2}$; and the ℓ_∞-norm as $\|\mathbf{x}\|_\infty = \max_{1 \leqslant i \leqslant n} |x_i|$. Finally, $\langle \cdot, \cdot \rangle$ and $\|\cdot\|$ denote respectively generic inner product and norm in a Hilbert space.

II. LEARNING THEORY BASICS FOR REGRESSION

This section provides a learning theoretic background in support of the following basic claim.

Given an atomic representation, different learning formulations introduce different structures in the materials

[a)] Electronic mail: quang.nguyen@epfl.ch

predictions. By choosing an appropriate learning formulation, we can optimize the relevant prediction metric.

A. Regression for atomization energies

We consider the following learning setting. Suppose that we observe a set of sample pairs $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ of different molecule representations $\mathbf{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ with the corresponding atomic energy $y_i \in \mathcal{Y} \subseteq \mathbb{R}$. Based on this data, we wish to estimate a function $f(\mathbf{x}) \to y$ for predicting atomization energy of new molecules. In what follows, we show that such an important feat is possible, given sufficient amount of training data.

It is important to quantify the quality of a predictor, which is often measured in terms of a test error computed over a test data $\{(\bar{\mathbf{x}}_j, \bar{y}_j)\}_{j=1}^N$. In practice, one typically considers three types of testing errors, MAE, MaxAE, and RMSE, which are defined as follows:

$$\text{MAE} = \frac{1}{l}\sum_{j=1}^{l} |f(\bar{\mathbf{x}}_j) - \hat{y}_j| = \frac{\|f(\bar{\mathbf{x}}) - \bar{\mathbf{y}}\|_1}{l},$$

$$\text{MaxAE} = \max_{j\in\{1,\dots,l\}} |f(\bar{\mathbf{x}}_j) - \hat{y}_j| = \|f(\bar{\mathbf{x}}) - \bar{\mathbf{y}}\|_\infty,$$

$$\text{RMSE} = \sqrt{\frac{1}{l}\sum_{j=1}^{l} |f(\bar{\mathbf{x}}_j) - \bar{y}_j|^2} = \sqrt{\frac{\|f(\bar{\mathbf{x}}) - \bar{\mathbf{y}}\|_2^2}{l}}.$$

Here $f(\bar{\mathbf{x}}) = [f(\bar{\mathbf{x}}_1), \dots, f(\bar{\mathbf{x}}_l)]^\top$ and $\bar{\mathbf{y}} = [\bar{y}_1, \dots, \bar{y}_l]^\top$.

The above metrics and their corresponding utilities are intuitive to the informed reader. For instance, RMSE metric looks at the average prediction error in the Euclidean distance, whereas MaxAE cares only about the worst case error. The metric MAE takes the other end of the spectrum and decreases the impact of outlier errors in the average as compared to RMSE. Hence, different applications may focus on any one of these prediction metrics.

The above regression framework can be categorized as a supervised learning problem and thus it has strong support from statistical learning theory[5–7], which will be introduced in what follows.

B. Supervised learning

In statistical learning theory, we typically assume that all the elements from the test data and the train data are independently and identically drawn according to a probability distribution (but one should keep in mind that the i.i.d. assumption can be further relaxed). We measure the performance of a function in terms of the expected loss/risk with respect to a loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$, see[6,7] for its precise definition. In what follows, for ease of presentation, we simply identify the expected risk as the test error over the test data $\{(\bar{\mathbf{x}}_i, \bar{y}_i)\}_{i=1}^l$, defined as

$$\mathcal{R}(f) = \frac{1}{l}\sum_{i=1}^{l} \ell(f(\bar{\mathbf{x}}_i), \bar{y}_i). \tag{1}$$

Such an argument will not cause any trouble when the test data size is sufficiently large. It is easy to see that MAE or RMSE is equivalent to $\mathcal{R}(f)$ with a suitable loss function. Note that MaxAE can not be directly linked to (1), but in the latter case, one can consider generalizing the definition of risk.

In this setting, one natural benchmark is the function $f^\star$ that minimizes the risk over all possible (i.e., measurable) functions. Often times, however, we have to restrict our search to some hypothesis space $\mathcal{F}$ of functions from $\mathbb{R}^d$ to $\mathbb{R}$ to exploit additional structures, such as smoothness, in the problem or to save on computation associated with the training procedure. The canonical example is the Kernel-based, linear prediction, the space of functions $f_\omega(\mathbf{x}) = \sum_{j=1}^p \omega_j \phi_j(\mathbf{x})$. Such an approach is also supported by many examples of consistency hypothesis spaces, i.e., $\inf_{f\in\mathcal{F}} \mathcal{R}(f) = \mathcal{R}(f^\star)$.[7] We will talk about how to choose a suitable hypothesis space in the later subsections.

C. Regularized M-estimators

With the given hypothesis space, a natural idea for finding a good predictor is to solve the expected risk minimization, $\inf_{f\in\mathcal{F}} \mathcal{R}(f)$. However, as the expected risk can not be known exactly, the expected risk minimization is replaced with the empirical risk minimization.

Directly minimizing the empirical loss can lead to an effect called overfitting, wherein we fit the training data extremely well (i.e., with low error), yet we obtain a model that produces very poor predictions on future test data whenever the test inputs differ from the training inputs. There exists an important solution to the overfitting problem, the regularized M-estimators[8] (also called as the regularized empirical risk minimizations[5]), i.e.,

$$\hat{f}_\lambda \in \operatorname*{argmin}_{f\in\mathcal{F}} \{\mathcal{R}_n^\lambda(f) := \mathcal{R}_n(f) + \lambda\Omega(f)\}. \tag{2}$$

Here, $\lambda \in \mathbb{R}_+$ is a regularization parameter, Ω is a regularizer and the empirical risk $\mathcal{R}_n(f)$ is defined as

$$\mathcal{R}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell\big(f(\mathbf{x}_i), y_i\big).$$

The regularizer Ω imposes certain properties on the underlying function. A very common property is the smoothness of the underlying function, which is especially required for performing atomization energies regression. Let's consider the case where f is represented via radial basis functions (RBFs) $\{\phi_k\}_{k\in\mathbb{N}}$, i.e.,

$$f(\mathbf{x}) = \sum_{k\in\mathbb{N}} \omega_k \phi_k(\mathbf{x}).$$

As radial basis functions are themselves smooth, imposing smoothness on f can be done by making the magnitude of weights $\|\omega\|^2 = \sum_{k\in\mathbb{N}} \omega_k^2$ be decayed. Weight decay implicitly leads to smoothness with RBF basis functions because rapid changes in the slope of f (i.e., high curvature) can only be created in RBFs by adding and subtracting basis functions with large weight. Considering a Hilbert space governed by RBFs, the magnitude of weight can be thought as the norm of f in this space. This type of regularizer is called ridge regularizer. In this paper, we mainly focus on ridge regularizer, but one should keep in mind that our approach still applies for a general regularizer.

In order to control the complexity of the solution and to ensure generalizing well, the regularization parameter λ needs to be tuned in practice. We will discuss this after presenting statistical results for the regularized M-estimators.

D. Statistical results

A key tool for analyzing statistical results for the regularized M-estimators is the error decomposition. To introduce the error decomposition, we introduce an auxiliary function f_λ, defined as the solution of the regularized expected risk minimization,

$$f_\lambda \in \operatorname*{argmin}_{f\in\mathcal{F}} \{\mathcal{R}^\lambda(f) := \mathcal{R}(f) + \lambda\Omega(f)\}.$$

A simple calculation shows that the excess risk of the estimator $\hat{f}_\lambda$ can be decomposed as (e.g.[9])

$$\mathcal{R}(\hat{f}_\lambda) - \mathcal{R}(f^\star) \leqslant \mathcal{E}_{\mathbf{sam}} + \mathcal{E}_{\mathbf{app}}, \tag{3}$$

where

$$\mathcal{E}_{\mathbf{sam}} = \mathcal{R}(\hat{f}_\lambda) - \mathcal{R}_n(\hat{f}_\lambda) + \mathcal{R}_n(f_\lambda) - \mathcal{R}(f_\lambda),$$

$$\mathcal{E}_{\mathbf{app}} = \mathcal{R}^\lambda(f_\lambda) - \mathcal{R}(f^\star).$$

We provide a proof for the above formulation in the appendix.

The first term $\mathcal{E}_{\mathbf{sam}}$ in the error decomposition above is a random variable depending on the training set, the function class $\mathcal{F}$, and the regularized parameter λ. It is called sample error. It measures the effect of minimizing the regularized empirical risk instead of the regularized expected risk. Typically, it can be controlled by a term which is decreasing with respect to both the train size and the regularization parameter λ.

The second term $\mathcal{E}_{\mathbf{app}}$ in the error decomposition above is deterministic. It only depends on the function class $\mathcal{F}$ and the regularization parameter λ. It measures how well the solution of the regularized expected risk minimization can be used to approximate $f^\star$. Typically, it is increasing with respect to the regularization parameter λ.

The regularization parameter λ hence controls an important trade-off in prediction performance (i.e., generalization), which has been extensively discussed in the literature[5,10]. An optimal trade-off based on the best choice of λ results in an excess risk that scales between the inverse and the inverse square root of the number of training data[8,11,12]. The following two examples provide statistical results for the estimators given by (2) with the square-norm penalty, considering two different learning problems.

Example II.1 Consider the setting of non-parametric regression with the square loss over a reproducing kernel Hilbert space (RKHS) $\mathcal{F}$ as those in[13,14]. It has been shown[15] that KRR, i.e., (2) with the square-norm penalty, has the following upper bounds for the excess risk,

$$\mathbb{E}[\mathcal{R}(\hat{f}_\lambda) - \mathcal{R}(f^\star)] \lesssim \frac{c_1}{n\lambda^\gamma} + c_2\lambda^{2\zeta}.$$

Here, $\gamma \in [0,1]$ is related to the capacity condition of $\mathcal{F}$ and $\zeta \in [1/2, 1]$ is related to the regularity of the target function $f^\star$. The optimal error bound $\mathcal{O}(n^{-\frac{2\zeta}{2\zeta+\gamma}})$ is achieved when $\lambda_* \simeq n^{-\frac{1}{2\zeta+\gamma}}$. The best choice of λ is depending on the unknown distribution parameters ζ and γ, and it is unknown. We thus choose the regularized parameter λ_* by using the cross-validation methods in practice.

Example II.2 Consider the setting of non-parametric classification over a RKHS $\mathcal{F}$ with the hinge loss as that in[11,12], or more general, a loss function with bounded gradient. Using the tools from Rademacher complexity[16], one can prove that the solution of (2) with the square-norm penalty has the following upper bounds on the excess risk:

$$\mathbb{E}[\mathcal{R}(\hat{f}_\lambda) - \mathcal{R}(f^\star)] \lesssim \frac{c_1}{\sqrt{n\lambda}} + c_2\lambda^\beta.$$

Here, we assume that the approximation error satisfies $\mathcal{E}_{\mathbf{app}} \lesssim \lambda^\beta$, for some $\beta \in (0,1]$. The best attainable error bound from the above estimates is of order $\mathcal{O}(n^{-\frac{\beta}{2\beta+1}})$, and it is achieved when $\lambda_* \simeq n^{-\frac{1}{2\beta+1}}$. Using a more involved technique, it has been shown in, e.g.[12], the error bound can be further improved to $\mathcal{O}(n^{-\alpha})$, where $\alpha \in (0,1]$ is a parameter depending on the data distribution and the hypothesis space $\mathcal{F}$.

E. Cross-validation methods

Unfortunately, the best choice of λ depends on the data distribution, and in practice, we have to use cross-validation (CV)[17]. In CV, we divide the training set into K roughly equal parts (called folds). For each k-th fold we fit the model with a candidate parameter λ to the other $K-1$ parts, using a specific algorithm for solving

(2) either with MAE, MaxAE or RMSE. This gives the regression coefficient $\mathbf{c}_k(\lambda)$ to compute the validation error $E_k(\lambda)$. The cross validation error is then computed as

$$\mathrm{CV}(\lambda) = \frac{1}{K}\sum_{k=1}^{K} E_k(\lambda),$$

and we choose λ_* to minimize the CV error.

F. Kernel trick

In this subsection, we discuss kernel methods, a common approach for atomization energies regression which is based on choosing the hypothesis space $\mathcal{F}$ as a RKHS generated by a kernel. The kernel trick is to map the original representation $\mathbf{x} \in \mathbb{R}^d$ to a representation in a Hilbert space $\mathcal{F}$, called the feature space, by a feature map $\psi\colon \mathbf{x} \mapsto \psi(\mathbf{x})$ in such a way that $\langle f, \psi(\mathbf{x})\rangle = f(\mathbf{x})$ for all function $f \in \mathcal{F}$. In other words, we map the data into a higher (possibly infinite) dimension space such that in this space, the predictor f can be determined by a linear expression. We then define the kernel function $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \langle \psi(\mathbf{x}), \psi(\mathbf{x}')\rangle$. One can think of $\mathcal{K}$ as specifying similarity between instances and of the feature map ψ as mapping the domain set $\mathcal{X}$ into a space where these similarities are realized as inner products. The main advantage of such trick is that it implements linear separators in high dimensional feature spaces without having to specify points in that space or expressing the feature map ψ explicitly. Problem (2) restricted to $\mathcal{F}$ now becomes

$$\hat{f}_\lambda \in \operatorname*{argmin}_{f\in\mathcal{F}} \ \frac{1}{n}\sum_{i=1}^{n} \ell\big(\langle f, \psi(\mathbf{x}_i)\rangle, y_i\big) + \lambda\Omega(f). \tag{4}$$

Although $\mathcal{F}$ can be infinite dimensional, solving (4) is equivalent to solving an optimization problem in finite-dimensional setting due to the following theorem.

Theorem II.3 (Representer theorem[18]) *Suppose that $\mathcal{F}$ is the feature space corresponding to the feature map ψ defined on $\mathcal{X}$. Then problem (4) with ridge regularizer possesses a solution of the following form*

$$\hat{f}_\lambda(\mathbf{x}) = \sum_{i=1}^{n} c_i^\natural \mathcal{K}(\mathbf{x}, \mathbf{x}_i),$$

where $\mathbf{c}^\natural = [c_1^\natural, \dots, c_n^\natural]^\top \in \mathbb{R}^n$.

Let the training matrix $\mathbf{K} = \big[\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)\big]_{1\leqslant i,j\leqslant n}$, and $\mathbf{K}_i$ be the i-th row of $\mathbf{K}$. Using the representer theorem, a simple calculation as shown in the appendix, we can see that solving the problem (4) with ridge regularizer is equivalent to solving the following optimization in finite dimensional setting

$$\min_{\mathbf{c}\in\mathbb{R}^n} \ \frac{1}{n}\sum_{i=1}^{n} \ell\Big(\mathbf{c}^\top\mathbf{K}_i, y_i\Big) + \frac{\lambda}{2}\mathbf{c}^\top\mathbf{K}\mathbf{c}. \tag{5}$$

Example II.4 (Two common kernels) Gaussian kernel: $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|_2^2/2\sigma^2)$. Laplacian kernel: $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|_1/\sigma)$. Intuitively, the Gaussian or Laplacian kernel sets the inner product in the feature space between $\mathbf{x}$ and $\mathbf{x}'$ to be zero if the instances are far away from each other (in the original domain) and close to 1 if they are close. The parameter σ and the corresponding norms determine what we mean by "close".

G. The SOAP-Average kernel

Different kernel such as Gaussian kernel or Laplacian kernel has been used widely in materials science community and has led to reasonable predictors. However, it is crucial to keep in mind that the way that the similarity between atomic configurations is measured will influence the quality of the predictor in kernel regressions. Smooth Overlap of Atomic Positions (SOAP)[4] based kernels[19,20] have been reported among the best performing kernels for predicting electronic structure properties of materials and molecules. For this paper, we will use the SOAP-Average kernel to measure the structural similarity between the molecules by combining the similarity measures of local environments.

Within the SOAP formalism, the local environment of the i-th atom within a molecule A, i.e., the abstract descriptor of the arrangement of atoms in its vicinity, will be denoted by $\mathcal{X}_i^A$. The set of all atoms of species α of molecule A is denoted by A^α. The local density of i-th atom of species α is then constructed as the superposition of Gaussian functions of variance σ^2 centered on this atom. A cutoff distance of r_c is imposed via a smooth function to set the size of the local environment.

$$\rho^\alpha_{\mathcal{X}_i^A}(\mathbf{r}) = \sum_{j\in A^\alpha} \exp\left(-\frac{(\mathbf{r}-\mathbf{r}_{ij})^2}{2\sigma^2}\right) f_{r_c}(|\mathbf{r}_{ij}|), \tag{6}$$

where $\mathbf{r}_{ij}$ is the Euclidean distance between atom i-th and atom j-th. The SOAP kernel is then defined as the overlap of two local atomic neighbor densities, integrated over the set $\mathbf{SO}(3)$ of all three dimensional rotations, as follow

$$\tilde{k}\big(\mathcal{X}_i^A, \mathcal{X}_j^B\big) = \int_{\mathbf{SO}(3)} \left| \sum_\alpha \int_{\mathbb{R}^3} \rho^\alpha_{\mathcal{X}_i^A}(\mathbf{r})\rho^\alpha_{\mathcal{X}_j^B}(\hat{R}\mathbf{r})\,\mathrm{d}\mathbf{r}\right|^2 \mathrm{d}\hat{R}. \tag{7}$$

In practice this kernel can be computed efficiently by first expressing the density on spherical harmonics basis [21]. The similarity measure $C_{ij}(A, B)$ between the local environments $\mathcal{X}_i^A$ and $\mathcal{X}_j^B$ of the molecules A and B is then determined by the SOAP average kernel function as

$$C_{ij}(A, B) = \frac{\tilde{k}\big(\mathcal{X}_i^A, \mathcal{X}_j^B\big)}{\sqrt{\tilde{k}\big(\mathcal{X}_i^A, \mathcal{X}_i^A\big)\tilde{k}\big(\mathcal{X}_j^B, \mathcal{X}_j^B\big)}}. \tag{8}$$

In order to extract a single similarity measure from the matrix of pairwise environment similarities $\mathbf{C}(A, B)$, the SOAP-Average[19] kernel combines the similarity information from the local kernels into a global similarity measure by taking average of all environment pair similarity values after raising them to the power ζ. As the normalized environment similarity values ranges between 0 and 1, 1 being identical: for value of $\zeta > 1$, higher similarity values naturally get higher weight in the averaged value.

$$\mathcal{K}_\zeta(A, B) = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} C_{ij}(A, B)^\zeta \tag{9}$$

where, N and M is the number of atoms in molecule A and B respectively. This kernel can be applied to both molecules and crystals while combining a detailed and systematic description of atomic structures with a large degree of adaptability through its hyper-parameters.

H. Multiple kernel learning

The choice of the kernel is critical to the success of the algorithm but in standard frameworks it is left to the user. While different kernels will lead to predictors with different qualities, all of them can be weighted to obtain a unique kernel as the input for classical kernel-based learning algorithms to get a much better predictor. This is called the problem of learning kernels in which one learns an optimal convex combination

$$\mathbf{K} = \sum_{j=1}^{m} \beta_j \mathbf{K}_j,\ \beta_j \geqslant 0,\ \sum_{j=1}^{m} \beta_j = 1,$$

of m given kernels $\mathbf{K}_1, \ldots, \mathbf{K}_m$. Methods to find the optimal weights were proposed in[22–24].

I. Error decomposition

The above subsections demonstrate that a solution for the minimization problem (5) with an appropriate regularization parameter λ and a suitable kernel has a good generalization performance. In general, Problem (5) is solved via an optimization procedure. Let $\hat{f}_{\lambda,\epsilon}$ be an ϵ-approximated solution of (5). A similar argument as that for (3), one can show that the statistical/generalization error of $\hat{f}_{\lambda,\epsilon}$ can be estimated as:

$$\mathcal{R}(\hat{f}_{\lambda,\epsilon}) - \mathcal{R}(f^\star) \leqslant \epsilon + \mathcal{E}_{\mathbf{sam}} + \mathcal{E}_{\mathbf{app}},$$

where

$$\mathcal{E}_{\mathbf{sam}} = \mathcal{R}(\hat{f}_{\lambda,\epsilon}) - \mathcal{R}_n(\hat{f}_\lambda) + \mathcal{R}_n(f_{\lambda,\epsilon}) - \mathcal{R}(f_\lambda),$$

$$\mathcal{E}_{\mathbf{app}} = \mathcal{R}^\lambda(f_\lambda) - \mathcal{R}(f^\star).$$

The term ϵ is called optimization error, while the other two terms are called as sample error and approximation error respectively. Similar estimations on $\mathcal{E}_{\mathbf{sam}}$ and $\mathcal{E}_{\mathbf{app}}$ as those in Subsection II D can be developed using tools from probability theory and approximation theory, which should be studied in the future. In the coming section, we focus on the optimization error, i.e., we study optimization procedures for solving (5).

III. CONVEX OPTIMIZATION OF ENERGIES REGRESSION

A. The basics of convex optimization

Statistical learning problem of molecules' energies regression explained in the previous section is modeled generically as the following composite convex optimization problem, considered as a sum a a data-fitting term and an explicit penalty term,

$$\Psi^\star := \min_{\mathbf{c} \in \mathbf{X} \subset \mathbb{R}^n} \{\Psi(\mathbf{c}) := g(\mathbf{c}) + h(\mathbf{M}\mathbf{c})\}, \tag{10}$$

where $\mathbf{X}$ is convex, $\mathbf{M}$ is $n \times m$ matrix, g and h are convex functions. While in most cases, finding an exact solution of (10) is impossible, we try to find an approximated solution, i.e., given a tolerance $\varepsilon > 0$, design methods in order to obtain $\mathbf{c} \in \mathbf{X}$ such that $\Psi(\mathbf{c}) - \Psi^\star \leqslant \varepsilon$.

Before review efficient numerical methods to approximate an optimal solution $\mathbf{c}^\star$ of (10) as well as required assumptions on h and g in the next sections, it is worthy to note that (10) covers the classical kernel ridge regression. Traditional approach to determine the regression coefficient $\mathbf{c}^\natural$ of a predictor f based on Representer theorem II.3 is to approximate it by the following

$$\mathbf{c}^\star = (\mathbf{K} + \lambda \mathbf{I}_n)^{-1} \mathbf{y}, \tag{KRR}$$

where $\mathbf{y} = [y_1, \ldots, y_n]^\top$ and $\mathbf{I}_n$ is the $n \times n$ identity matrix. Simple calculations show that $\mathbf{c}^\star$ is a solution to the following ridge-regularized least square minimization problem, considered as a particular instance of (10),

$$\min_{\mathbf{c} \in \mathbb{R}^n} \frac{1}{2} \|\mathbf{K}\mathbf{c} - \mathbf{y}\|_2^2 + \frac{\lambda}{2} \mathbf{c}^\top \mathbf{K} \mathbf{c}. \tag{ℓ_2}$$

This optimization problem however might have more than one solution than $\mathbf{c}^\star$.

Being inspired by the interests in MAE and MaxAE within the materials science community, we are looking for new kernel-based models that could improve these metrics while still keep RMSE in the same order of magnitude as (KRR). This can be done by exploring the inner structure of these metrics. For instance, to adapt the MAE, instead of using ℓ_2-loss function, we propose the use of ℓ_1-loss function.

Example III.1 (Ridge ℓ_1-loss regression) In this model, we estimate regression coefficient $\mathbf{c}^\natural$ as follows

$$\min_{\mathbf{c} \in \mathbb{R}^n} \|\mathbf{K}\mathbf{c} - \mathbf{y}\|_1 + \frac{\lambda}{2} \mathbf{c}^\top \mathbf{K} \mathbf{c}. \tag{ℓ_1}$$

Our second model deploys the ℓ_∞-norm loss function to adapt the structure of MaxAE.

Example III.2 (Ridge ℓ_∞-loss regression) We estimate regression coefficient $\mathbf{c}^\natural$ as follows

$$\min_{\mathbf{c}\in\mathbb{R}^n} \|\mathbf{K}\mathbf{c}-\mathbf{y}\|_\infty + \frac{\lambda}{2}\mathbf{c}^\top\mathbf{K}\mathbf{c}. \qquad (\ell_\infty)$$

Both (ℓ_1) and (ℓ_∞) lie beyond the effective methods of linear algebra and smooth optimization and we need deeper numerical methods to approximate their solutions. Most common ones are first-order methods. These methods obtain reasonable accuracy numerical solutions by using only first-order oracle information from the objective, such as gradient estimates. They can also handle the non-smooth variants by making use of the proximal mapping principle. Main advantages of these methods are their scalability and nearly dimension-independent convergence rates. Coupled with recent demand for low-to-medium accuracy solutions in applications, these methods indeed provide a critical trade-off between the complexity-per-iteration and the iteration-convergence rate along with the ability to distribute and parallelize computation.

Assumption 1. We assume that ∇g is L_g-Lipschitz continuous, i.e.,

$$(\forall \mathbf{c}_1\in\mathbb{R}^n)(\forall \mathbf{c}_2\in\mathbb{R}^n) \quad \|\nabla g(\mathbf{c}_1)-\nabla g(\mathbf{c}_2)\| \leqslant L_g\|\mathbf{c}_1-\mathbf{c}_2\|.$$

B. A primal first-order method

Generally, h can be non-smooth and hence we will need more efforts to deal with (10) due to the presence of the matrix $\mathbf{M}$. However, in the case when this matrix is identity, i.e.,

$$\Psi^\star := \min_{\mathbf{c}\in\mathbb{R}^n}\{\Psi(\mathbf{c}) := h(\mathbf{c})+g(\mathbf{c})\}, \qquad (11)$$

where g satisfies Assumption 1, we can solve it efficiently by different versions of proximal-gradient method. Its motivation is to use the linear approximation of smooth g and simply include the nonsmooth term h in an explicit fashion

$$\mathbf{c}^{k+1} = \operatorname*{argmin}_{\mathbf{c}\in\mathbb{R}^n} g(\mathbf{c}^k)+\nabla g(\mathbf{c}^k)^\top(\mathbf{c}-\mathbf{c}^k)+\frac{1}{2\alpha_k}\|\mathbf{c}-\mathbf{c}^k\|_2^2+h(\mathbf{c}), \qquad (12)$$

with the step-size $\alpha_k \leqslant 1/L_g$. This optimization problem is the update rule of the proximal-gradient method

$$\mathbf{c}^{k+1} = \mathrm{prox}_{\alpha_k h}(\mathbf{c}^k-\alpha_k\nabla g(\mathbf{c}^k)), \qquad (13)$$

where the proximal operator is defined as

$$\mathrm{prox}_h(\bar{\mathbf{c}}) = \operatorname*{argmin}_{\mathbf{c}\in\mathbb{R}^n} h(\mathbf{c})+\frac{1}{2}\|\mathbf{c}-\bar{\mathbf{c}}\|_2^2. \qquad (14)$$

If we simply set the step-size $\alpha_k = 1/L_g$ in the proximal gradient method, we will get the following convergence rate

$$\Psi(\mathbf{c}^N)-\Psi^\star = \mathcal{O}\big(N^{-1}\big), \qquad (15)$$

which implies that in order to obtain an ε-approximated solution, we need $\mathcal{O}(\varepsilon^{-1})$ iterations. This rate can be upgraded to $\mathcal{O}(N^{-2})$ by making use of an extra-momentum step and hence $\mathcal{O}(\varepsilon^{-1/2})$ iterations will be needed to get an ε-approximation. The full version is the following:

Algorithm 1 Fast iterative shrinkage-thresholding algorithm (FISTA)[25]

1: Inputs: $\hat{\mathbf{c}}^1 = \mathbf{c}^0 \in \mathbb{R}^n$, $t_1 = 1$.
2: **for** $k = 1, 2, \ldots, N-1$ **do**

$$\mathbf{c}^k = \mathrm{prox}_{\alpha_k h}(\hat{\mathbf{c}}^{\mathbf{k}} - \alpha_k\nabla g(\hat{\mathbf{c}}^{\mathbf{k}})),$$
$$t_{k+1} = 0.5\left(1+\sqrt{1+4t_k^2}\right)$$
$$\hat{\mathbf{c}}^{k+1} = \mathbf{c}^k + \frac{t_k-1}{t_{k+1}}\big(\mathbf{c}^{\mathbf{k}}-\mathbf{c}^{k-1}\big).$$

3: **end for**
4: **return** $\mathbf{c}^N$.

Example III.3 (Proximal operators) . Given $\gamma > 0$ and $\bar{\mathbf{c}} \in \mathbb{R}^n$.

1. Proximal operator of ℓ_1-norm $\|\cdot\|_1$: Solving

$$\operatorname*{argmin}_{\mathbf{c}\in\mathbb{R}^n} \|\mathbf{c}\|_1 + \frac{1}{2\gamma}\|\mathbf{c}-\bar{\mathbf{c}}\|_2^2$$

is equivalent to solving following n convex problems in dimension 1

$$\min_{c\in\mathbb{R}} |c| + \frac{1}{2\gamma}|c-\bar{c}|^2.$$

Elementary computations shows that this problem has an analytical solution given by $\tau_\gamma(\bar{c})$, where τ is the shrinkage operator defined by $\tau_\gamma(c) = (|\bar{c}|-\gamma)_+\,\mathrm{sign}(\bar{c})$.

2. Proximal operator of ℓ_∞-norm $\|\cdot\|_\infty$: Solving

$$\hat{\mathbf{c}} = \operatorname*{argmin}_{\mathbf{c}\in\mathbb{R}^n} \|\mathbf{c}\|_\infty + \frac{1}{2\gamma}\|\mathbf{c}-\bar{\mathbf{c}}\|_2^2$$

can be done as follow:

(a) Compute $\tilde{\mathbf{c}}$, the projection of $\gamma^{-1}\bar{\mathbf{c}}$ onto the unit ball $\{\|\mathbf{c}\|_1 \leqslant 1\}$[26].
(b) Apply[27] to obtain $\hat{\mathbf{c}} = \bar{\mathbf{c}} - \gamma\tilde{\mathbf{c}}$.

C. A primal-dual first-order method

When incorporating with a non-identity matrix $\mathbf{M}$ in (10), the proximal-gradient methods can not be used.

In these cases, a primal-dual method is necessary. This method and its versions are based on the representing h using its Fenchel conjugate function h^* which is defined as

$$h^*(\mathbf{d}) = \sup_{\mathbf{c}\in\mathbb{R}^n} \mathbf{c}^\top\mathbf{d} - h(\mathbf{c}).$$

Using Fenchel conjugaison, we can represent h as follow

$$h(\mathbf{Mc}) = \sup_{\mathbf{d}\in\mathbb{R}^n} (\mathbf{Mc})^\top\mathbf{d} - h^*(\mathbf{d}).$$

One of the methods used to constructs approximations to a solution to (10) is the following

Algorithm 2 Accelerated Primal-Dual method[28]

1: Inputs: $\mathbf{c}^1 \in \mathbb{R}^n$, $\mathbf{d}^1 \in \mathbb{R}^n$, $\mathbf{c}^1_{ag} = \mathbf{c}^1$, $\mathbf{d}^1_{ag} = \mathbf{d}^1$, $\bar{\mathbf{c}}^1 = \mathbf{c}^1$.
2: **for** $k = 1, 2, \ldots, N-1$ **do**

$$\begin{aligned}
\mathbf{c}^k_{md} &= (1-\beta_k^{-1})\mathbf{c}^k_{ag} + \beta_k^{-1}\mathbf{c}^k,\\
\mathbf{d}^{k+1} &= \mathrm{prox}_{\tau_k h^*}(\mathbf{d}^k - \mathbf{M}\bar{\mathbf{c}}_k),\\
\mathbf{c}^{k+1} &= \mathbf{c}^k - \eta_k(\nabla g(\mathbf{c}^k_{md}) + \mathbf{M}^\top\mathbf{d}^{k+1}),\\
\mathbf{c}^{k+1}_{ag} &= (1-\beta_k^{-1})\mathbf{c}^k_{ag} + \beta_k^{-1}\mathbf{c}^{k+1},\\
\mathbf{d}^{k+1}_{ag} &= (1-\beta_k^{-1})\mathbf{d}^k_{ag} + \beta_k^{-1}\mathbf{d}^{k+1},\\
\bar{\mathbf{c}}^{k+1} &= \theta_k(\mathbf{c}^{k+1} - \mathbf{c}^k) + \mathbf{c}^{k+1}.
\end{aligned}$$

3: **end for**
4: **return** $\mathbf{c}^N_{ag}$ and $\mathbf{d}^N_{ag}$.

Like the proximal-gradient method, the primal-dual method requires the computation of proximal operator of Fenchel conjugate function which can be deduced from the proximal operator of the original function itself due to[27]. Simply set $\beta_k = \frac{k+1}{2}$, $\theta_k = \frac{k-1}{k}$, $\eta_k = \frac{3k}{4\eta}$ and $\tau_k = \frac{1}{\eta}$ for $\eta = 2L_g + 2\|\mathbf{M}\|(N-1) + \frac{N\sqrt{13(N-1)}}{2\tilde{D}}$ with $\tilde{D} > 0$, the rate that a primal-dual method can achieve is only

$$\Psi(\mathbf{c}^N_{ag}) - \Psi^\star = \mathcal{O}\big(N^{-1}\big).$$

IV. NUMERICAL EXPERIMENTS

A. GDB9 data set

GDB9 data set[29] consisting of chemical representations and the internal energies U_0 (Hartree) at absolute zero temperature of 133884 small organic molecules. We divide this data set into two parts: training set contains 100000 molecules and testing set contains 33884 molecules. Building upon the data from recently published paper[20] discussing the SOAP-Average kernel based predictions for this database, three kernel matrices $\mathbf{K}_2$, $\mathbf{K}_3$ and $\mathbf{K}_4$ were computed on training set using (9) with three different local environment sizes, $r_c \in \{2, 3, 4\}$ Angstrom, aiming to capture the atomic interactions at different scales. These three kernels matrices are then weighted as follow

$$\mathbf{K} = \frac{256}{273}\mathbf{K}_2 + \frac{16}{273}\mathbf{K}_3 + \frac{1}{273}\mathbf{K}_4.$$

We use portions of this matrix to predict U_0 for 33884 molecules in testing set using (KRR), (ℓ_1) and (ℓ_∞).

B. Computational complexity

As (KRR) requires an inverse operation of an $n \times n$-matrix and the multiplication of an $n \times n$-matrix and an n-coordinates vector, its best known complexity is $\mathcal{O}(n^{2.373})$. Generally, both models (ℓ_1) and (ℓ_∞) with SOAP-Average kernels could be solved numerically by standard optimization approaches such as Algorithm 2. The complexity of this method is the complexity of the computation of proximal operator. In our case of soft-threshold, its complexity is only $\mathcal{O}(n)$ and hence the overall complexity of the method is $\mathcal{O}(nN)$ with N is the total number of iterations.

Being stated that primal-dual method iteratively builds an approximation solution with the rate of $\mathcal{O}(1/N)$, relatively slow in particular for such an ill-conditioned kernel like SOAP-Average kernel. In order to accelerate the convergence's speed, we propose two techniques concerning models (ℓ_1) and (ℓ_∞).

Preconditioning. Main idea is replacing it by a small perturbation parameter ρ, i.e., setting $\mathbf{M} = \mathbf{K} + \rho\mathbf{I}_n$ and then making the change of the variable $\mathbf{d} = \mathbf{Mc}$ to reformulate (ℓ_1) as

$$\min_{\mathbf{d}\in\mathbb{R}^n} \|\mathbf{d} - \mathbf{y}\|_1 + \frac{\lambda}{2}\mathbf{d}^\top\mathbf{M}^{-1}\mathbf{d}, \tag{16}$$

while (ℓ_∞) becomes

$$\min_{\mathbf{d}\in\mathbb{R}^n} \|\mathbf{d} - \mathbf{y}\|_\infty + \frac{\lambda}{2}\mathbf{d}^\top\mathbf{M}^{-1}\mathbf{d}. \tag{17}$$

These problems can be solved efficiently and quickly by FISTA (Algorithm 1) with the overall complexity $\mathcal{O}(n^{2.273} + nN)$. However, within a convergence rate of $\mathcal{O}(1/N^2)$ of FISTA, our approach will reach a solution in predefined tolerance within reasonable small total iterations N, and hence it does require a computation cost not much more than (KRR).

Dual formulation. Another trick to avoid matrix inverse calculation is using the dual formulation of (16) and (17), says

$$\min_{\mathbf{d}\in\mathbb{R}^n} g^*(\mathbf{d}) + \frac{1}{2\lambda}\mathbf{d}^\top\mathbf{K}\mathbf{d}, \tag{18}$$

where g^* is Fenchel conjugate function of either $\|\cdot - \mathbf{y}\|_1$ for (ℓ_1) or $\|\cdot - \mathbf{y}\|_\infty$ for (ℓ_∞). This dual formulation can be solved efficiently by FISTA. By duality[27], the original coefficient is then recovered by $\mathbf{c}^N = -\lambda^{-1}\mathbf{d}^N$. We note that the computational complexity of the proximal operator of g^* is the same as that of of g. The total computational complexity in this case is only $\mathcal{O}(nN)$.

C. Cross-validation

The tuning parameter λ is selected by using 10-folds cross-validation by screening 15 values on a base-10 logarithmic grid from 10^{-9} to 10^{0}. This procedure will be parallelized.

D. Simulation results

Different training sets 100, 500, 1000, 5000, 10000, 25000, extracted from training set of 100000 molecules, are used to predict U_0 using (KRR), (ℓ_1) and (ℓ_∞) for molecules in testing set and then compute the metrics: MAE, MaxAE and RMSE. The results, recorded in Table I, show that (ℓ_1) improves significantly MAE in comparison with (KRR) while (ℓ_∞) achieves a better MaxAE than (KRR). The bold values are the best accuracies.

	KRR/ℓ_2	ℓ_1	ℓ_∞
training size = 100			
MAE	8.16	8.16	8.16
MaxAE	75.65	75.68	75.68
RMSE	10.86	10.86	10.86
training size = 500			
MAE	3.76	**3.22**	4.79
MaxAE	68.30	49.56	**54.00**
RMSE	5.01	4.37	6.26
training size = 1000			
MAE	2.50	**2.06**	2.47
MaxAE	61.83	51.13	**49.58**
RMSE	3.47	2.97	3.42
training size = 5000			
MAE	1.10	**0.92**	2.25
MaxAE	62.08	59.38	**41.78**
RMSE	1.60	1.39	2.92
training size = 10000			
MAE	0.72	**0.59**	0.76
MaxAE	69.18	61.58	**45.04**
RMSE	1.14	1.03	1.12
training size = 25000			
MAE	0.40	**0.34**	0.40
MaxAE	45.53	48.54	**44.39**
RMSE	0.74	0.82	0.73

TABLE I. Prediction errors (kcal/mol).

V. CONCLUSIONS

We present novel settings together with efficient algorithms to approximate a numerical solution for molecules' energies prediction via statistical learning theory. We also perform numerical advantages of our approach with two new models: ridge ℓ_1-loss minimization and ridge ℓ_∞-loss minimization.

ACKNOWLEDGEMENTS

This research was supported by the NCCR MARVEL, a National Centres of Competence in Research, funded by the Swiss National Science Foundation.

REFERENCES

[1] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," Physical review **136**, B864 (1964).
[2] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," Physical review **140**, A1133 (1965).
[3] D. J. Wright, G. Capon, R. Page, J. Quiroga, A. A. Taseen, and F. Tomasini, "Evaluation of forecasting methods for decision support," International journal of forecasting **2**, 139–152 (1986).
[4] A. P. Bartók, R. Kondor, and G. Csányi, "On representing chemical environments," Phys. Rev. B **87**, 184115 (2013).
[5] V. N. Vapnik and V. Vapnik, *Statistical learning theory*, Vol. 1 (Wiley New York, 1998).
[6] F. Cucker and D. X. Zhou, *Learning theory: an approximation theory viewpoint*, Vol. 24 (Cambridge University Press, 2007).
[7] I. Steinwart and A. Christmann, *Support vector machines* (Springer Science & Business Media, 2008).
[8] S. A. Geer, *Empirical Processes in M-estimation*, Vol. 6 (Cambridge university press, 2000).
[9] O. Bousquet and L. Bottou, "The tradeoffs of large scale learning," in *Advances in neural information processing systems* (2008) pp. 161–168.
[10] S. Boucheron, O. Bousquet, and G. Lugosi, "Theory of classification: A survey of some recent advances," ESAIM: probability and statistics **9**, 323–375 (2005).
[11] T. Zhang, "Statistical behavior and consistency of classification methods based on convex risk minimization," Annals of Statistics , 56–85 (2004).
[12] I. Steinwart and C. Scovel, "Fast rates for support vector machines using gaussian kernels," The Annals of Statistics , 575–607 (2007).
[13] S. Smale and D.-X. Zhou, "Learning theory estimates via integral operators and their approximations," Constructive approximation **26**, 153–172 (2007).
[14] A. Caponnetto and E. De Vito, "Optimal rates for the regularized least-squares algorithm," Foundations of Computational Mathematics **7**, 331–368 (2007).
[15] J. Lin and V. Cevher, "Optimal rates for spectral-regularized algorithms with least-squares regression over Hilbert spaces," arXiv:1801.07226 (2018).
[16] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," Journal of Machine Learning Research **3**, 463–482 (2002).
[17] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, Vol. 14 (Stanford, CA, 1995) pp. 1137–1145.
[18] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," The annals of statistics , 1171–1220 (2008).
[19] S. De, A. P. Bartók, G. Csányi, and M. Ceriotti, "Comparing molecules and solids across structural and alchemical space," Physical Chemistry Chemical Physics **18**, 13754–13769 (2016).
[20] A. P. Bartók, S. De, C. Poelking, N. Bernstein, J. R. Kermode, G. Csányi, and M. Ceriotti, "Machine learning unifies the modeling of materials and molecules," Science Advances **3** (2017), 10.1126/sciadv.1701816, http://advances.sciencemag.org/content/3/12/e1701816.full.pdf.
[21] A. P. Bartók, M. J. Gillan, F. R. Manby, and G. Csányi, "Machine-learning approach for one- and two-body corrections to density functional theory: Applications to molecular and condensed water," Phys. Rev. B **88**, 054104 (2013).

[22]S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," Journal of Machine Learning Research **7**, 1531–1565 (2006).
[23]H. Kadri, A. Rakotomamonjy, F. Bach, and P. Preux, "Multiple operator-valued kernel learning," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'12 (Curran Associates Inc., USA, 2012) pp. 2429–2437.
[24]H. Kadri, E. Duflos, P. Preux, S. Canu, A. Rakotomamonjy, and J. Audiffren, "Operator-valued kernels for learning from functional response data," The Journal of Machine Learning Research **17**, 613–666 (2016).
[25]A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," SIAM journal on imaging sciences **2**, 183–202 (2009).
[26]J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the l_1-ball for learning in high dimensions," in *Proceedings of the 25th international conference on Machine learning* (ACM, 2008) pp. 272–279.
[27]H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, Vol. 408 (Springer, 2011).
[28]Y. Chen, G. Lan, and Y. Ouyang, "Optimal primal-dual methods for a class of saddle point problems," SIAM J. Optim. **24**, 1779–1814 (2014).
[29]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).

APPENDIX

In this appendix, we provide the proofs for some of the elementary inequalities in this paper.

A. Proof of (3)

We have

$$\begin{aligned}\mathcal{R}(\hat{f}_\lambda) - \mathcal{R}(f^\star) &\leqslant \mathcal{R}^\lambda(\hat{f}_\lambda) - \mathcal{R}(f^\star)\\ &= [\mathcal{R}(\hat{f}_\lambda) - \mathcal{R}_n(\hat{f}_\lambda) + \mathcal{R}_n(f_\lambda) - \mathcal{R}(f_\lambda)]\\ &\quad + [\mathcal{R}_n^\lambda(\hat{f}_\lambda) - \mathcal{R}_n^\lambda(f_\lambda)] + [\mathcal{R}^\lambda(f_\lambda) - \mathcal{R}(f^\star)]\\ &\leqslant [\mathcal{R}(\hat{f}_\lambda) - \mathcal{R}_n(\hat{f}_\lambda) + \mathcal{R}_n(f_\lambda) - \mathcal{R}(f_\lambda)]\\ &\quad + [\mathcal{R}^\lambda(f_\lambda) - \mathcal{R}(f^\star)]\\ &= \mathcal{E}_{\mathbf{sam}} + \mathcal{E}_{\mathbf{app}},\end{aligned}$$

where for the inequality, we used the fact that $\mathcal{R}_n^\lambda(\hat{f}_\lambda) \leqslant \mathcal{R}_n^\lambda(f_\lambda)$ since $\hat{f}_\lambda$ is a solution for (2).

B. Proof for (5)

Due to the representer theorem, instead of solving optimization problem (4) with respect to functions f, we find the weights of the predictor, a vector $\mathbf{c}^\natural$ in a finite dimensional space. For $\hat{f}_\lambda(\mathbf{x}) = \sum_{i=1}^n c_i^\natural \mathcal{K}(\mathbf{x}, \mathbf{x}_i)$, we have that for all i,

$$\begin{aligned}\ell(\hat{f}_\lambda(\mathbf{x}_i), y_i) &= \ell\bigg(\sum_{j=1}^n c_j^\natural \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j), y_i\bigg)\\ &= \ell\bigg(\sum_{j=1}^n c_j^\natural \mathbf{K}_{ij}, y_i\bigg) = \ell\Big((\mathbf{c}^\natural)^\top \mathbf{K}_i, y_i\Big),\end{aligned}$$

and since

$$\begin{aligned}\Big\langle \hat{f}_\lambda, \psi(\mathbf{x})\Big\rangle &= \hat{f}_\lambda(\mathbf{x}) = \sum_{i=1}^n c_i^\natural \mathcal{K}(\mathbf{x}, \mathbf{x}_i)\\ &= \sum_{i=1}^n c_i^\natural \left\langle \psi(\mathbf{x}), \psi(\mathbf{x}_i)\right\rangle = \left\langle \psi(\mathbf{x}), \sum_{i=1}^n c_i^\natural \psi(\mathbf{x}_i)\right\rangle,\end{aligned}$$

we have $\hat{f}_\lambda = \sum_{i=1}^n c_i^\natural \psi(\mathbf{x}_i)$ and hence,

$$\begin{aligned}\|\hat{f}_\lambda\|^2 &= \left\langle \sum_{i=1}^n c_i^\natural \psi(\mathbf{x}_i), \sum_{i=1}^n c_i^\natural \psi(\mathbf{x}_i)\right\rangle\\ &= \sum_{i,j=1}^n c_i^\natural c_j^\natural \left\langle \psi(\mathbf{x}_i), \psi(\mathbf{x}_j)\right\rangle\\ &= \sum_{i,j=1}^n c_i^\natural c_j^\natural \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)\\ &= (\mathbf{c}^\natural)^\top \mathbf{K}\mathbf{c}^\natural.\end{aligned}$$

The problem (4) with ridge regularizer now becomes

$$\min_{\mathbf{c}\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^n \ell\Big(\mathbf{c}^\top \mathbf{K}_i, y_i\Big) + \frac{\lambda}{2}\mathbf{c}^\top \mathbf{K}\mathbf{c}.$$