PADOVA UNIVERSITY PRESS

Dolomites Research Notes on Approximation

Special Issue dedicated to Robert Schaback on the occasion of his 75th birthday, Volume 15 · 2022 · Pages 125–145

Mean-field limits of trained weights in deep learning: A dynamical systems perspective

Alexandre Smirnov[a] · Boumediene Hamzi[b] · Houman Owhadi[c]

Communicated by Gabriele Santin

Abstract

Training a residual neural network with L^2 regularization on weights and biases is equivalent to minimizing a discrete least action principle and to controlling a discrete Hamiltonian system representing the propagation of input data across layers. The kernel/feature map analysis of this Hamiltonian system suggests a mean-field limit for trained weights and biases as the number of data points goes to infinity. The purpose of this paper is to investigate this mean-field limit and illustrate its existence through numerical experiments and analysis (for simple kernels).

1 Introduction

Supervised learning is a class of learning problems which seek to find a relationship between a set of predictor variables (the inputs) and a set of target variables (the outputs). The target, also called dependent variable, can correspond to a quantitative measurement (e.g., height, stock prices, etc.) or to a qualitative measurement (e.g., sex, class, etc.). The problem of predicting quantitative variables is called regression. On the other hand, classification problems aim at predicting qualitative variables. Over many years of research, various models have been developed to solve both type of problems. In particular, *artificial neural network* (ANN) models have become very popular since their complex architecture allows the model to capture underlying patterns in the data. This architecture has proved to be successful in different areas of artificial intelligence such as image processing, speech recognition or text mining. ANNs transform the inputs using a composition of consecutive mappings, generally represented by a directed acyclic graph. With this representation, the inputs are being propagated through multiple layers, defining a trajectory that is optimized by training the parameters of the network.

In this paper we investigate the propagation of the inputs from a Dynamical Systems perspective, an approach recently explored by Houman Owhadi in [10]. Owhadi shows that the minimizers of L_2 regularized ResNets, a particular class of ANNs, satisfy a discrete least action principle implying the near preservation of the norm of weights and biases across layers. The parameters of trained ResNets can be identified as solutions of a Hamiltonian system defined by the activation function and the architecture of the ANN. The form of the Hamiltonian suggests that it is amenable to mean-field limit analysis as the number of data points increases. Furthermore, when the mean-field limit holds, the trajectory of inputs across layers is given by a mean-field Hamiltonian system where position and momentum variables are nearly decoupled through mean-field averaging. **The purpose of this paper is to analyse and numerically illustrate this mean-field limit by investigating the convergence of the trained weights and biases of the model (appearing in the Hamiltonian in our kernel/feature map setting) as the number of data points goes to infinity.**

In Section A we define the mathematical setting of supervised learning and introduce kernel learning, a class of problems that encompasses ANNs and which is used throughout this paper. In Section B we explain how to identify the optimization of neural networks as a dynamical system problem and characterize the optimal trajectory of the inputs. In Section 2 we reformulate the dynamics using the feature space representation of kernels. With this representation we show that the trajectory is determined by a parameter amenable to mean-field limit analysis. In Section 3, we simulate this parameter on regression and classification datasets.

All the results in the Appendices are a review of some results in Owhadi's paper [10].

2 Mean-field analysis of Hamiltonian dynamics

In Section B, we derived the Hamiltonian representation of minimizers of mechanical regression and idea registration. We stated that the propagation of the inputs is completely determined by the initial momentum $p(0)$. In this section, we derive

[a] Department of Mathematics, Imperial College London, United Kingdom, email: alexandre.smirnov20@imperial.ac.uk
[b] Department of Computing and Mathematical Sciences, Caltech, CA, USA.email:boumediene.hamzi@gmail.com
[c] Department of Computing and Mathematical Sciences, Caltech, CA, USA. email: owhadi@caltech.edu

another system which does not involve the momentum. The system is obtained for an operator-valued kernel Γ with feature map $\psi : \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{X}, \mathcal{F})$ such that (42) holds,

$$\Gamma(x, x') = \psi^T(x)\psi(x').$$

Then, the ODE for the trajectory is of the form

$$\begin{cases} \dot{q}_i = \psi^T(q_i)\alpha(t) \\ q_i(0) = x_i, \end{cases} \tag{1}$$

for some mapping $\alpha(t)$. We characterize α using the feature map identification (44) of the RKHS $\mathcal{V}$ and show that this mapping is amenable to mean-field limit analysis. We also discuss a data-based method to simulate this limit.

2.1 Feature space representation of kernels

We reformulate mechanical regression and idea registration using the feature space representation of kernels (cf., Sec. A.4) as presented in [10, Sec. 6].

2.1.1 Mechanical regression

The following theorem characterizes the minimizers of mechanical regression in the feature space of Γ.

Theorem 2.1 (**Mechanical regression in feature space**). *Let Γ be a kernel of RKHS $\mathcal{V}$ with associated feature space and map $\mathcal{F}$ and $\psi : \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{X}, \mathcal{F})$. Then, $\alpha_1, \dots, \alpha_L$ satisfy*

$$\begin{cases} \textit{Minimize} & \frac{\nu L}{2} \sum_{s=1}^{L} \|\alpha_s\|_{\mathcal{F}}^2 \; + \; \ell(\phi_L(X), Y), \\ \textit{over} & \alpha_1, \dots, \alpha_L \in \mathcal{F}, \end{cases} \tag{2}$$

if and only if the $v_s(\cdot) = \psi^T(\cdot)\alpha_s \in \mathcal{V}$ minimize (59) and

$$\phi_L(\cdot) = (I + \psi^T(\cdot)\alpha_L) \circ \cdots \circ (I + \psi^T(\cdot)\alpha_1). \tag{3}$$

This theorem identifies the minimizers v_s with parameters α_s belonging to the feature space $\mathcal{F}$ (possibly infinite-dimensional). In practice, we choose Γ to be a scalar operator-valued kernel (see Def. A.4) obtained with a finite dimensional feature space $\mathfrak{F}$ and map φ,

$$\Gamma(x, x') = \varphi^T(x)\varphi(x')I_{\mathcal{X}}.$$

Using Lemma A.4, $\{\alpha_s\}_{s=1}^L$ are matrices in $\mathbb{R}^{\dim(\mathcal{X})\times\dim(\mathfrak{F})}$ and $\|\alpha_s\|_{\mathcal{F}}^2$ corresponds to a matrix Frobenius norm. These parameters define the optimal discrete trajectory $q^1, \dots, q^{L+1}$

$$\begin{cases} q^{s+1} = q^s + \alpha_s \varphi(q^s), \;\; \text{for } s = 1, \dots, L \\ q^1 = X, \end{cases} \tag{4}$$

where we used $\psi^T(\cdot)\,\alpha_s = \alpha_s\varphi(\cdot)$. By introducing $\Delta t = 1/L$ and $\alpha_s = \Delta t\,\tilde{\alpha}_s$, the system is equivalent to

$$\begin{cases} q^{s+1} = q^s + \Delta t\,\tilde{\alpha}_s \varphi(q^s), \;\; \text{for } s = 1, \dots, L \\ q^1 = X, \end{cases} \tag{5}$$

where $\{\tilde{\alpha}_s\}_{s=1}^L$ minimize

$$\frac{\nu}{2} \sum_{s=1}^{L} \|\tilde{\alpha}_s\|_{\mathcal{F}}^2 \Delta t \; + \; \ell(q^{L+1}(X), Y), \text{ where } q^{L+1} = \phi_L(X). \tag{6}$$

(5) is a discrete dynamical system which approximates (1) (for each input x_i). The time-dependent parameter $\alpha(t)$ is approximated by $\{\alpha_s\}$ with $\alpha_s = \alpha(s/L)$ at time $t_s = s/L$.

2.1.2 Idea registration

The next theorem characterizes α in the feature map representation of idea registration.

Theorem 2.2 (**Idea registration in feature space**). *The $\alpha(t)$ satisfy*

$$\begin{cases} \textit{Minimize} & \frac{\nu}{2} \int_0^1 \|\alpha(t)\|_{\mathcal{F}}^2 \, dt \; + \; \ell(\phi^\nu(X, 1), Y) \\ \textit{over} & \alpha \in C([0, 1], \mathcal{F}) \end{cases} \tag{7}$$

if and only if $v(\cdot, t) = \psi^T(\cdot)\alpha(t)$ and $\phi^\nu(x, t)$ minimize (69). Furthermore, at the minimum, $\|\alpha(t)\|_{\mathcal{F}}^2$ is constant over $t \in [0, 1]$.

Hence, α is identified as the minimizer of (7) and (71) implies $\dot{q}_i = \psi^T(q_i)\alpha(t)$. If in addition, Γ is scalar operator-valued, the ODE for the trajectory becomes

$$\begin{cases} \dot{q}_i = \alpha(t)\varphi(q_i) \\ q_i(0) = x_i. \end{cases} \tag{8}$$

2.1.3 Equivalence with ResNet block minimizers

In Sec. 2.1.1 we reformulated mechanical regression using the feature map representation of Γ. In addition, if Γ is a SOV kernel, the minimizers are identified with matrices $\{\alpha_s\}_{s=1}^L$. In the special case that Γ and K have the form in (54), mechanical regression is equivalent to minimizing one ResNet block (see Ex. A.3) with L_2 regularization on weights and biases. This is summarized in the following theorem.

Theorem 2.3. *If $\Gamma(x,x') = \varphi(x)^T\varphi(x')I_{\mathcal{X}}$ and $K(x,x') = \varphi(x)^T\varphi(x')I_{\mathcal{Y}}$, where $\varphi(x)$ is given by (53), then minimizers of (59) are of the form*

$$f(x) = \tilde{\alpha}\varphi(x) \text{ and } v_s(x) = \alpha_s\varphi(x), \tag{9}$$

where $\tilde{\alpha} = (\tilde{W}, \tilde{b}) \in \mathcal{L}(\mathcal{X}\oplus\mathbb{R}, \mathcal{Y})$, $\alpha_s = (W_s, b_s) \in \mathcal{L}(\mathcal{X}\oplus\mathbb{R}, \mathcal{X})$ are minimizers of

$$\min_{\tilde{\alpha},\alpha_1,\dots,\alpha_L} \frac{\nu L}{2}\sum_{s=1}^{L}\|\alpha_s\|^2_{\mathcal{L}(\mathcal{X}\oplus\mathbb{R},\mathcal{X})} + \lambda\|\tilde{\alpha}\|^2_{\mathcal{L}(\mathcal{X}\oplus\mathbb{R},\mathcal{Y})} + \|f(\phi_L(X)) - Y\|^2_{\mathcal{Y}^N}, \tag{10}$$

and $\|\alpha\|_{\mathcal{L}(\mathfrak{F},\mathcal{Z})}$ corresponds to the Frobenius norm of the linear map $\alpha : \mathfrak{F} \longrightarrow \mathcal{Z}$.

The approximation $f^\ddagger = f\circ\phi_L$ is of the form

$$f\circ\phi_L(x) = (\tilde{\alpha}\varphi)\circ(I+\alpha_L\varphi)\circ\cdots\circ(I+\alpha_1\varphi)(x). \tag{11}$$

2.2 Mean-field analysis

2.2.1 The mean-field approximation

Mean-field theory aims at developing approximation strategies for systems composed of interacting particles. It was originally developed in statistical physics to study systems such as the Ising model [2]. The central idea behind mean-field theory is to approximate the interacting terms in the system by a simpler non-interacting function. Our aim is to apply this strategy to study the Hamiltonian (79),

$$\mathfrak{H}(q,p) = \frac{1}{2}\sum_{i,j=1}^{N} p_i^T\Gamma(q_i,q_j)p_j. \tag{12}$$

It can be viewed as a system of interacting particles. The feature map representation of Γ allows us to split the interacting term $\Gamma(q_i,q_j)$ into two terms $\psi^T(q_i)$ and $\psi(q_j)$. If we re-scale the momentum $\bar{p}_j \coloneqq Np_j$ and define

$$\alpha(t) \coloneqq \frac{1}{N}\sum_{j=1}^{N}\psi(q_j(t))\bar{p}_j(t), \tag{13}$$

we can remove the interaction in the Hamiltonian

$$\mathfrak{H}(q,p) = \frac{1}{2}\sum_{i=1}^{N} p_i^T\psi^T(q_i)\alpha(t).$$

We can rewrite the dynamics of (q,p) as

$$\begin{cases}\dot{q}_i = \psi^T(q_i)\alpha\\ \dot{\bar{p}}_i = -\partial_x\left(\bar{p}_i^T\psi^T(x)\alpha\right)\Big|_{x=q_i}\end{cases}. \tag{14}$$

Note that this coincides with the dynamics of q obtained in Theorem 2.2. Eq. (13) suggests that $\alpha(t)$ is amenable to mean-field limit analysis as $N\to\infty$. If the limit exists, the system (12) behaves like a decoupled system of particles in the sense that the trajectory of each data input is not influenced by other inputs. Secondly, $\alpha(t)$ defines the ODE for the flow map ϕ^v (70),

$$\dot{\phi}^v(x,t) = \alpha(t)\,\phi^v(x,t)$$

which determines the solution of idea registration $f^\ddagger(\cdot) = f(\phi^v(\cdot,1))$.

2.2.2 Type of convergence

In theory, $\alpha(t)$ is a possibly infinite-dimensional object belonging to the feature space $\mathcal{F}$ of the operator-valued kernel Γ. In practice, we consider Γ to be scalar operator-valued defined with a finite-dimensional feature space $\mathfrak{F} = \mathbb{R}^D$. Lemma A.4 states that $\alpha(t)\in\mathbb{R}^{p\times D}$ for all $t\in[0,1]$ (taking $\mathcal{Y}=\mathcal{X}$ in the lemma). To simulate the limit, we need to plot a functional of $\{\alpha(t)\}_{0<t<1}$ against different values of N. A natural choice corresponds to the energy integral appearing in the loss (7),

$$E(\alpha) = \int_0^1 \|\alpha(t)\|^2_{\mathcal{F}}\,dt. \tag{15}$$

Numerically, we approximate $\alpha(t)$ with $\{\alpha_s\}_{s=1}^L$ using mechanical regression. The energy (15) is then approximated by

$$\begin{aligned} E(\alpha_1,\dots,\alpha_L) &= \sum_{s=1}^{L} \|\alpha_s\|_{\mathcal{F}}^2 \,\Delta t \\ &= \frac{1}{L}\sum_{s=1}^{L} \|\alpha_s\|_{\mathcal{F}}^2, \end{aligned} \tag{16}$$

which corresponds to the average Frobenius norm of α_s across the layers of the network, and appears in the formulation (6) of mechanical regression.

2.3 Numerical simulations

Provided that Γ is a Scalar Operator Valued (SOV) kernel, the feature space representation of mechanical regression given by Theorem 2.1 reduces the search for the optimal trajectory to the search for matrices $\{\alpha_s\}_{s=1}^L$ that minimize (2). These parameters correspond to a discretization of the time-dependent parameter $\alpha(t)$ in idea registration (Thm. 2.2). We present the optimization algorithm with the feature space framework.

Let $\mathcal{X}=\mathbb{R}^p$ and $\mathcal{Y}=\mathbb{R}^d$ be the input and output space. Let Γ and K be scalar operator-valued kernels obtained using feature maps φ, φ_2 with corresponding feature spaces $\mathfrak{F}$ and $\mathfrak{F}_2$,

- $\Gamma(x,x') = \varphi^T(x)\varphi(x')I_{\mathcal{X}}$, where $\varphi(x)\in\mathfrak{F}=\mathbb{R}^D$ for some $D\geq 1$.
- $K(x,x') = \varphi_2^T(x)\varphi_2(x')I_{\mathcal{Y}}$, where $\varphi_2(x)\in\mathfrak{F}_2=\mathbb{R}^{D_2}$ for some $D_2\geq 1$.

Obtain $\phi_L(X)=q^{L+1}$ from the discrete dynamics (4). The function f is of the form $f(\cdot)=w_2\varphi_2(\cdot)$, where

$$w_2 = \left(\varphi_2^T(\phi_L(X))\varphi_2(\phi_L(X)) + (N\lambda)I_{D_2}\right)^{-1}\varphi_2^T(\phi_L(X))\,Y \in \mathbb{R}^{d\times D_2} \tag{17}$$

minimizes the ridge loss

$$\ell_{ridge}(\phi_L(X),Y) = \frac{1}{N}\|w_2\varphi_2(\phi_L(X)) - Y\|^2 + \lambda\|w_2\|_{\mathcal{F}_2}^2, \tag{18}$$

where the squared error term has been normalized. By renormalizing the squared error in the ridge loss term, **we aim to illustrate the convergence of the trained model parameters $\{\alpha_s\}_{s=1}^L$ and w (obtained as minimizers of (2)) as $N\to\infty$.**

The normalization of the squared error will control the ridge loss as N increases. If the trained model parameters $\{\alpha_s\}_{s=1}^L$ and w (obtained as minimizers of (2)) converge as $N\to\infty$, then the energy integral $E(\alpha)$ must converge too.

For any deformed input $\phi_L(X)$, (17) uniquely defines w_2. Thus, the loss (2) is completely determined by $\alpha_1,\dots,\alpha_L$,

$$L(\alpha_1,\dots,\alpha_L) := \frac{\nu L}{2}\sum_{s=1}^{L}\|\alpha_s\|_{\mathcal{F}}^2 + \ell_{ridge}(\phi_L(X),Y). \tag{19}$$

We can use a gradient-based method to minimize the loss (19) with respect to $\alpha_1,\dots,\alpha_L$. In this article we use the Adam optimizer, a gradient-based method that computes adaptive learning rates for each parameter [12].

The feature perspective has the advantage that it doesn't involve simulating a mechanical system unlike the shooting method (Sec. B.4).

2.4 Analytical example: Linear kernel

In order to gain intuition about the optimization of $\alpha(t)$, we derive an analytical solution to (1) when $\mathcal{X}=\mathbb{R}$ and Γ is the scalar operator-valued linear kernel

$$\Gamma(x,x') = \varphi(x)\varphi(x')I_{\mathcal{X}},\quad \varphi(x)=x.$$

Applying Ex. A.5 with $p=1$, the feature space of Γ is $\mathcal{F}=\mathbb{R}$. The trajectory (8) is

$$\begin{aligned} &\dot{q}_i = \alpha(t)q_i \\ \implies &\frac{\dot{q}_i}{q_i} = \alpha(t) \\ \implies &q_i(t) = x_i\exp\left(\int_0^t \alpha(s)\,ds\right). \end{aligned}$$

Energy preservation of $\|\alpha(t)\|_{\mathcal{F}}^2 = \alpha^2(t)$ over t implies that $\alpha(t)$ is constant in t, i.e. $\alpha(t)\equiv\alpha\in\mathbb{R}$. Hence, the inputs are deformed according to the trajectory

$$q_i(t) = x_i\,e^{\alpha t} \implies q_i(1) = x_i e^{\alpha}.$$

The deformation depends on the optimal α that minimizes the loss (7), which can be explicitly written as,

$$\begin{aligned} L(\alpha) &= \frac{\nu}{2}\alpha^2 + \lambda Y^T[K(Xe^{\alpha}, Xe^{\alpha}) + \lambda I_N]^{-1}Y \\ &= \frac{\nu}{2}\alpha^2 + \lambda e^{-2\alpha} Y^T[K(X,X) + \lambda e^{-2\alpha} I_N]^{-1}Y. \end{aligned}$$

The minimizer α of L changes the regularization λ on the ridge loss into a parameter $\tilde{\lambda} = \lambda e^{-2\alpha}$. Therefore, deforming the input space is equivalent to tuning the regularization of the ridge loss. Since $\tilde{\lambda} \to 0$ as $\alpha \to \infty$, the regularization $\nu > 0$ penalizes overfitting of the training data.

3 Simulations in feature space representation

In this Section we simulate the parameter $\alpha(t)$ described in Section 2 for both regression and classification datasets. For regression, one is a synthetic dataset and the second is the Boston housing dataset [4]. For classification datasets, we choose MNIST and Fashion MNIST, two benchmark sets.

3.1 Model optimization

The input and output spaces are $\mathcal{X} = \mathbb{R}^p$ and $\mathcal{Y} = \mathbb{R}^d$.

3.1.1 Choice of feature maps

We suppose that the kernels Γ, K are scalar operator-valued kernels obtained using feature maps φ, φ_2 with corresponding feature spaces $\mathfrak{F}$ and $\mathfrak{F}_2$ defined as follows:

- $\Gamma(x,x') = \varphi(x)^T\varphi(x')I_{\mathcal{X}}$, where $\varphi(x) = (\boldsymbol{a}(x), 1) \in \mathfrak{F} = \mathbb{R}^{p+1}$ is defined through an activation as in Sec. A.4.3.
- $K(x,x') = \varphi_2(x)^T\varphi_2(x')I_{\mathcal{Y}}$, where $\varphi_2(x) = \boldsymbol{a}(W^2x + b^2) \in \mathfrak{F}_2$, $W^2 \in \mathbb{R}^{\dim\mathfrak{F}_2 \times p}$ and $b_2 \in \mathbb{R}^{\dim\mathfrak{F}_2}$. We take $\mathfrak{F}_2 = \mathbb{R}^{D_2}$. The entries $W^2_{i,j}, b^2_i$ are taken randomly as discussed in Sec. A.4.4.

3.1.2 Choice of other parameters

We train the model for three different activation functions $\boldsymbol{a}(x)$: ReLU, tanh and sigmoid. The kernel Γ is used to deform the input space and its feature map up-samples the inputs by one dimension. For the ridge kernel K we can choose $\mathfrak{F}_2$ such that it up-samples or down-samples the inputs. For our experiments we fix $\mathfrak{F}_2 = \mathbb{R}^{10}$.

3.1.3 Model training and comparison

We train the model as described in Sec. 2.3 using the Adam optimizer to minimize the L_2 regularized loss (19). We initialize the parameters at $\alpha_s = 0$ for $s = 1, \dots, L$, which corresponds to no deformation of the inputs, i.e. $q^{L+1} = X$. We apply ridge regression without space deformation, then we plot the training and tests errors. For regression data, the MSE is used as the metric error; for classification we use the proportion of misclassified samples. We show test errors for the sake of completeness since the purpose of this paper is not to improve test error but to **study the convergence of the trained model parameters as we increase** N. To analyze convergence, we plot the energy $E(\alpha)$ given by (16) as a function of N, for different activation functions and different values of L.

3.2 Regression datasets

3.2.1 Synthetic data

We take a synthetic dataset $(X_i, Y_i)_{i=1}^N$ where $X_i \in \mathcal{X} = \mathbb{R}$ and

$$Y_i = X_i^2 + X_i + \epsilon_i, \;\; \epsilon_i \sim \mathcal{N}(0, \sigma^2) \;\text{ for } \sigma^2 = 1.$$

We choose N training inputs evenly spaced in the interval $[-5, 5]$ with $N = 50 \cdot 2^k$ and $k = 0, 1, \dots, 5$. We fix the test dataset to 200 points: 100 in each of the intervals $[-7.5, -5]$ and $[5, 7.5]$.

3.2.2 Boston housing dataset

The Boston housing data [4] was collected in 1978 and each of the $N = 506$ entries represent aggregated data about 14 features for homes from various suburbs in Boston, Massachusetts. The target variable is the median value of owner-occupied homes and the remaining features are the predictors. Thus, we have $\mathcal{X} = \mathbb{R}^{13}$ and $\mathcal{Y} = \mathbb{R}$. For reasons of numerical instability, we only train the model with ReLU and tanh activations.

We split the dataset into training and test set, where we choose 10% of the total set size for test data (around 50 points), the remaining data is used for training. Test set size is kept low on purpose since we are interested in getting as much training data to get the most accurate estimate of mean-field convergence.

3.2.3 Result interpretation

Model performance

On the synthetic data (Fig. 4), mechanical regression outperforms ridge regression for any choice of L and activation, except for tanh and $L = 2$. For ReLU and tanh using $L = 2$ layers produces a higher training/test MSE and L_2 regularized loss compared to deeper models; for $L \geq 4$, the models have very similar MSE and objective loss Fig. (4A), Fig. (4(B). For the sigmoid activation, these values are identical for all values of L Fig. (4C). The key element is the deformation of the inputs, which improves both the fit and the L_2 regularized loss: compare the Ridge curves and mechanical regression curves. For all activations, the deformation yields a considerable reduction in L_2 loss.

On the Boston housing dataset (Fig. (9)), mechanical regression outperforms ridge regression on the training set for both activation functions. The best decrease in MSE on the training set is for the ReLU activation (9A).

Convergence

On the synthetic dataset, Fig. 5 shows that $E(\alpha)$ appears to converge as N increases. This seems to holds for of all values of L. The exceptions are mechanical regression with $L = 2$ and the ReLU activation (5A), which is upward trending, as well as with sigmoid activation.
Fig. 6 shows that on the Boston housing dataset, $E(\alpha)$ fluctuates less as L increases. Hence, using a deeper network, controls the energy as we increase N. When $L = 2$ fluctuations are very important compared to other values of L.
Finally, an increase in L appears to reduce the average norm $E(\alpha)$ on both datasets. This means that the energy in the loss (19), becomes smaller as we decrease the time step used to discretize idea registration.

3.3 Classification datasets

3.3.1 MNIST

MNIST is a classification dataset containing images of handwritten digits. It has a training set of 60,000 examples and a test set of 10,000 examples. We train mechanical regression with N data points taken from the training set, where $N \in \{500, 1000, \ldots, 5000\}$ with a step of 500. We choose $\mathfrak{F}_2 = \mathbb{R}^{784}$, which corresponds to the dimension of a flattened 28×28 input image. The database is widely used for training and testing models in machine learning.

3.3.2 Fashion MNIST

Fashion MNIST is a dataset of Zalando's article images consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28×28 image, associated with a label from 10 classes. Observe that it shares the same image size and structure of training and testing splits as MNIST. This dataset is used for benchmarking machine learning algorithms and is regarded as more complicated than MNIST (in terms of model performance).

3.3.3 Result interpretation

Model performance

On MNIST (Fig. 13), mechanical regression with ReLU and tanh converge towards a $5-6\%$ testing error (2a,2b) as we increase the number of training data points, while sigmoid converges towards a 14% testing error (2c). The same pattern holds for Fashion MNIST (Fig. 18), with ReLU and tanh converging towards $15-16\%$ and sigmoid towards 21%. Interestingly, as N increases, the training accuracy decreases, but test accuracy increases. A possible explanation is that increasing N, increases the penalty in the ridge loss ($N\lambda$ in Eq. (17)) which prevents the model from overfitting. This can lead to better generalization, which is reflected in the better test accuracy and worse training accuracy.

Convergence

Fig. 14 (MNIST) and Fig. 19 (Fashion MNIST) have very similar shapes. Convergence clearly holds for $L = 2, 3$. For $L = 1$, convergence seems to be slower since the values are slightly decreasing in N. The curves are flatter for MNIST which indicates a faster convergence than for Fashion MNIST. This could be explained by the fact that Fashion MNIST is a more challenging classification problem than MNIST, which requires more training samples.

3.4 Discussion

3.4.1 Observations

Our model presented in Sec. 3.1 uses the feature map representation of mechanical regression to approximate idea registration. We train the model on several classification and regression datasets, including a dynamical system. We observed the behaviour of the energy $E(\alpha)$ after minimizing the loss for increasing values of N. On the synthetic, MNIST and Fashion MNIST datasets we can observe clear convergence in N. On MNIST and Fashion MNIST, there is also convergence of the test loss. On the Boston housing dataset, the fluctuation of the energy decreases as we use more hidden layers but we cannot observe a similar convergence to MNIST.

3.4.2 Limitations and suggestions

The rationale behind initializing α to 0 is the following: since the regularization ν induces a penalty on the deformation of the inputs, we allow it to occur only if the reduction in ridge loss is greater than this penalty. The limitation of this reasoning is that the optimizer may get stuck at a local minimum, hence not exploring the full parameter space. One approach to overcome this could be to optimize α with different random initializations and pick the one that minimizes the loss (19).

Secondly, the high-dimensionality of α complicates the study of element-wise convergence. We use the energy since its presence in the loss function makes it a natural candidate. Other transformations can be explored.

Thirdly, the value of $E(\alpha)$ is certainly affected by the penalty parameters λ and ν. We have not explored other values which can change it. For example, an interesting case would be to try the simulations for $\nu = 0$, thus not penalizing the deformation.

Finally, other network architectures will lead to different values for $E(\alpha)$ and possibly faster convergence to a minimum. Hence, further experiment with other model architectures should be performed.

4 Conclusion

In this article we presented a dynamical system perspective of the training of neural networks by characterizing the optimal propagation of the inputs across its layers. We reviewed the theory of kernel based learning and introduced operator-valued kernels as a tool to construct spaces of functions. Using this setting, we introduced a model architecture, called *mechanical regression*, whose minimizers are identified as a solution of a discrete Hamiltonian system. We obtained the continuous dynamics as an infinite-depth limit of the discrete case.

We then used the feature space representation of kernels to reformulate the dynamics of the minimizers. The form of the trajectory in this representation is amenable to mean-field limit analysis. We showed a data-based approach to simulate a functional of this limit, involving the training of a neural network. Finally, we trained our model on various standard datasets and investigated its performance and the limiting behaviour of the functional. We observed a convergence in energy on MNIST and Fashion MNIST, two benchmark classification datasets. However, we should verify this convergence with other datasets and model architectures which motivates the need for further investigation.

5 Acknowledgment

Parts of this work were done when B. H. was a Marie Curie fellow at Imperial College London. B. H. thanks the European Commission for funding through the Marie Curie fellowship STALDYS-792919 (Statistical Learning for Dynamical Systems). H. O. gratefully acknowledges support by the Air Force Office of Scientific Research under award number FA9550-18-1-0271 (Games for Computation and Learning).

6 Code

The Python codes of all numerical experiments in this paper are at https://github.com/alexsm98/Mean-field-limit-code-.git

A Supervised learning with Kernels

A.1 Introduction

Supervised learning aims at finding a relationship between $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. We are given N pairs of training data points $\{(x_i, y_i)\}_{i=1}^N$, which we use to model the relationship. We set $X = (x_1, \dots, x_N) \in \mathcal{X}^N$ and $Y = (y_1, \dots, y_N) \in \mathcal{Y}^N$. In regression, $\mathcal{X} = \mathbb{R}^p$, where the p coordinates are called features, and $\mathcal{Y} = \mathbb{R}^d$ for some $p, d \geq 1$. In classification, $\mathcal{Y}$ corresponds to a discrete subset $\mathbb{D} = \{c_1, \dots, c_l\} \subset \mathbb{R}^d$. We formulate the supervised learning problem as follows.

Problem A.1 (**Supervised learning**). Let $f^\dagger$ be an unknown function mapping $\mathcal{X}$ to $\mathcal{Y}$. Given the information $f^\dagger(X) = Y$ with the training data $(X, Y) \in \mathcal{X}^N \times \mathcal{Y}^N$ approximate $f^\dagger$ with a mapping f that minimizes a loss function

$$\ell \ : \ \mathcal{Y}^N \times \mathcal{Y}^N \longrightarrow \mathbb{R}_{\geq 0}.$$

In parametric models, we look for the optimal f in a certain Hilbert space of functions $\mathcal{H}$ and its performance is evaluated using a given metric. It is important to note the difference between *loss* and *metric*. The former is the objective function that we minimize to get f, while the latter measures the model's performance on the data. In regression, the *mean squared error* (MSE)

$$\ell_{\mathcal{Y}}(f(X), Y) := \frac{1}{N}\|f(X) - Y\|^2_{\mathcal{Y}^N}, \ \text{ where } \ \|Y\|^2_{\mathcal{Y}^N} := \sum_{i=1}^N \|Y_i\|^2_{\mathcal{Y}},$$

is usually chosen as the metric, while in classification data we use the proportion of misclassified samples. The MSE can also be used as a loss function for both regression and classification. Throughout this paper we will use the *ridge loss*

$$\ell_{ridge}(f(X), Y) := \|f(X) - Y\|^2_{\mathcal{Y}^N} \ + \ \lambda \|f\|^2_{\mathcal{H}}, \tag{20}$$

where $\lambda > 0$ is a positive regularization parameter and $\|f\|^2_{\mathcal{H}}$ is the norm of the function f in the Hilbert space $\mathcal{H}$ to which it belongs. This loss introduces an additional penalty term $\lambda > 0$ on f to prevent overfitting of the training data, an issue that commonly arises when using the MSE as the loss. Overfitting means that the model well-approximates the training output Y but fails to generalize to new data, resulting in large approximation error.

The ridge loss can be used for both regression and classification. In regression, the output Y is directly predicted with $\hat{Y} = f(X)$. In classification, $f(X)$ is a vector whose dimension is equal to the number of possible classes. The class is predicted with $\hat{Y} = \arg\max f(X)$.

Example A.1 (**Linear ridge regression**)**.** We aim to determine a parametric function of the form $f_{\boldsymbol{\theta}}$

$$\begin{aligned} f_{\boldsymbol{\theta}} \ &: \mathbb{R}^p \longrightarrow \mathbb{R} \\ &x \longmapsto f_{\boldsymbol{\theta}}(x) = x^T \boldsymbol{\theta} \end{aligned}$$

The Hilbert space $\mathcal{H}$ can be identified with the space of parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ and the penalty term regularizes the Euclidean norm $\|f_{\boldsymbol{\theta}}\|^2_{\mathcal{H}} = \|\boldsymbol{\theta}\|^2$. The parameter that minimizes the ridge loss is

$$\boldsymbol{\theta}_{ridge} = \left(X^T X + \lambda I_N\right)^{-1} X^T Y. \tag{21}$$

A new input x^* is predicted as

$$y^* = \boldsymbol{\theta}^T_{ridge}\, x^* = \sum_{k=1}^{p} \theta_k\, x^*_k.$$

The solution also admits a dual form, obtained using the relationship (cf., [13])

$$\left(X^T X + \lambda I_N\right)^{-1} X^T Y = X^T \left(X X^T + \lambda I_N\right)^{-1} Y. \tag{22}$$

Thus, we can write $\boldsymbol{\theta}_{ridge} = X^T \boldsymbol{\alpha}$ and the prediction as

$$y^* = \sum_{i=1}^{N} \alpha_i\, x_i^T x^*. \tag{23}$$

When formulating the problem in dual form, we optimize over a parameter $\boldsymbol{\alpha} \in \mathbb{R}^N$, whose dimension corresponds to the number of data points. Additionally, observe the presence of dot product terms $x_i^T x^*$, which can be viewed as a similarity measure between two input samples. Kernel methods generalize this idea by replacing the dot product with a function $k(x_i, x^*)$ which corresponds to an inner product of the inputs in a transformed space, called the *feature space*.

A.2 Neural networks

In this section, we introduce neural networks and refer the reader to [3] for a more detailed overview on this topic.

A *neural network* can be used as a supervised learning model whose architecture contains multiple layers of input data transformation. The general idea is to transform the input x into x' by a composition of consecutive mappings. We then map the modified input to the output y. The function f can be expressed as a composition

$$f = f_D \circ \cdots \circ f_1, \tag{24}$$

where $f_k : \mathcal{X}_k \longrightarrow \mathcal{X}_{k+1}$ and f_k belongs to a Hilbert space $\mathcal{H}_k$ for $k = 1, \ldots, D$. We have $\mathcal{X}_0 = \mathcal{X}, \mathcal{X}_{D+1} = \mathcal{Y}$. Note that each function f_k maps $\mathcal{X}_k$ into a new space $\mathcal{X}_{k+1}$. The space $\mathcal{X}_k$ is commonly referred to as the k-th layer of the network and all the layers $\mathcal{X}_k \neq \mathcal{X}, \mathcal{Y}$ are called the hidden layers.

Example A.2 (**Artificial Neural Network**)**.** Each function f_k is of the form

$$f_k(x) = \boldsymbol{a}(W_k\, x \,+\, b_k), \tag{25}$$

where $\boldsymbol{a}$ is a fixed activation function, $W_k \in \mathcal{L}(X_k, X_{k+1})$, and $\mathcal{L}$ is the space of linear operators between X_k and X_{k+1}, and $b_k \in X_{k+1}$. Popular choices for $\boldsymbol{a}$ are the Relu, tanh or sigmoid activations. The function W_k is a bounded linear operator called the weight of the k-th layer; the term b_k is called the bias. With this architecture the solution to Pb. A.1 is the minimizer of the loss

$$\min_{W_k, b_k} \ell(f(X), Y).$$

In practice, we have $\mathcal{X}_k = \mathbb{R}^{d_k}$ for all k, such that W_k is a matrix in $\mathbb{R}^{d_k \times d_{k+1}}$.

Example A.3 (**Residual Neural Network**)**.** Write $f = F_D \circ \cdots \circ F_1$, where

$$F_k = f_k \circ (I + v^k_{L_k}) \circ \cdots \circ (I + v^k_1), \tag{26}$$

where f_k is defined as in Example A.2 and $v^k_s = \boldsymbol{a}(W^k_s\, x \,+\, b^k_s)$. We can also take $v^k_s = W^k_s \boldsymbol{a}(x) \,+\, b^k_s$. Residual neural networks are commonly called ResNets.

More generally, one can choose any sequence of Hilbert spaces $\mathcal{H}_1, \ldots, \mathcal{H}_D$ to which the mappings $f_1, \ldots, f_D$ belong. In particular, we will choose *Reproducing Kernel Hilbert* (RKHS) spaces, which are defined by a kernel function. We present the theory of kernel learning and RKHS in the next section.

A.3 Scalar-valued Kernels

In this section, we provide a brief summary of supervised learning with scalar real-valued kernels. For a more detailed overview we suggest [14] and [15].

A.3.1 Motivation and the kernel trick

The idea behind kernel methods is to map the input space $\mathcal{X}$ into a Hilbert space $(\mathcal{F}, \langle\cdot,\cdot\rangle_{\mathcal{F}})$ with a mapping $\Phi : \mathcal{X} \longrightarrow \mathcal{F}$ and look for linear models in this space,

$$f_{\boldsymbol{\theta}}(x) = \langle \boldsymbol{\theta}, \Phi(x)\rangle_{\mathcal{F}}.$$

In view of (23) and assuming that $\mathcal{F} = \mathbb{R}^P$, an input x^* is predicted as

$$y^* = \sum_{i=1}^{N} \alpha_i \, \Phi^T(x_i)\, \Phi(x^*), \tag{27}$$

where

$$\alpha = \left(\Phi(X)\Phi^T(X) + \lambda\, I_N\right)^{-1} Y \tag{28}$$

and the matrix $\Phi(X)$ has rows $\Phi^T(x_i)$. Eq. (27),(28) use the feature space only to calculate inner products since $(\Phi(X)\Phi^T(X))_{i,j} = \Phi^T(x_i)\Phi(x_j)$. Hence, if we define a function $k : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$, called a *positive definite kernel* (or simply kernel), such that

$$k(x_i, x_j) = \langle\, \Phi(x_i), \Phi(x_j)\rangle_{\mathcal{F}}, \tag{29}$$

the model in (27) becomes

$$y^* = \sum_{i=1}^{N} \alpha_i \, k(x_i, x^*).$$

The approximation f is

$$f(\cdot) = \sum_{i=1}^{N} \alpha_i \, k(x_i, \cdot). \tag{30}$$

Eq. (29) corresponds to the definition of a real-valued kernel $k : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ (see [5]). Thus, a kernel corresponds to an inner product in a Hilbert space, called the feature space. This approach is called the *kernel trick* and the theoretical result that justifies (30) is the *Representer theorem* presented in Sec. A.3.3.

Recall that supervised learning aims at finding the best approximation f within a Hilbert space of functions. Eq. (30) suggests that this space is defined by the kernel k. That is indeed the case and the space is the *Reproducing Kernel Hilbert Space* (RKHS) of k.

A.3.2 Reproducing Kernel Hilbert Space

We start with the following definitions [15, p. 119].

Definition A.1. Let $\mathcal{H}$ be a Hilbert space of functions $f : \mathcal{X} \longrightarrow \mathbb{R}$ defined on a non-empty set $\mathcal{X}$.
(i) A function $k : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ is called a ***reproducing kernel*** of $\mathcal{H}$ if we have $k(\cdot, x) \in \mathcal{H}$ for all $x \in \mathcal{X}$ and the ***reproducing property***

$$f(x) = \langle f, k(x,\cdot)\rangle \tag{31}$$

holds for all $f \in \mathcal{H}$ and all $x \in \mathcal{X}$. In particular,

$$\langle k(x,\cdot), k(x',\cdot)\rangle = k(x,x'). \tag{32}$$

(ii) The space $\mathcal{H}$ is called a ***reproducing kernel Hilbert space*** over $\mathcal{X}$ if for all $x \in \mathcal{X}$ the Dirac functional $\delta_x : \mathcal{H} \longrightarrow \mathbb{R}$ defined by

$$\delta_x(f) := f(x), \;\; f \in \mathcal{H},$$

is continuous.

From (32) it follows that k is symmetric in its arguments and satisfies the conditions for positive definiteness, hence it is a kernel. In fact, every RKHS has a unique reproducing kernel k which spans $\mathcal{H}$ as follows,

$$\mathcal{H} = \overline{\text{span}\{k(x,\cdot) \,|\, x \in \mathcal{X}\}}.$$

On the other hand, given a kernel k we want to define an RKHS such that k is its (unique) reproducing kernel. Thm. 4.21 in [15, p. 121] states that every kernel admits a unique RKHS $\mathcal{H}$, which can be defined as the completion of the set

$$\mathcal{H}_{pre} := \left\{ \sum_{i=1}^{n} \alpha_i k(x_i,\cdot) \,:\, n \in \mathbb{N},\, \alpha_1,\ldots,\alpha_n \in \mathbb{R},\, x_1,\ldots,x_n \in \mathcal{X} \right\}.$$

For every $f = \sum_{i=1}^{n} \alpha_i \, k(x_i,\cdot) \in \mathcal{H}_{pre}$ we have

$$\|f\|^2_{\mathcal{H}} = \sum_{i,j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \tag{33}$$

Consequently, we have a bijection between kernels and RKHS of functions.

A.3.3 The Representer Theorem

Given a training dataset $(X,Y) \in \mathcal{X}^N \times \mathcal{Y}^N$ we consider the supervised learning problem, where the space of functions is the RKHS $\mathcal{H}$ of a kernel k. The following theorem provides the explicit form of the solution to this problem [14, p.90] (with slight modifications in the notations).

Theorem A.1 (**Representer theorem**). *Define*

1. $\Omega : [0,\infty) \longrightarrow \mathbb{R}$ *a strictly increasing monotonic function.*
2. $\ell_{\mathcal{Y}} : \left(\mathcal{Y}^N \times \mathcal{Y}^N\right) \longrightarrow \mathbb{R} \cup \{\infty\}$ *an arbitrary error function.*

This defines a loss function

$$\ell(f) = \ell_{\mathcal{Y}}(f(X),Y) \; + \; \Omega(\|f\|). \tag{34}$$

Then a minimizer of ℓ in $\mathcal{H}$ admits a representation of the form

$$f(\cdot) = \sum_{i=1}^{N} \alpha_i k(x_i,\cdot). \tag{35}$$

Example A.4 (**Kernel ridge regression**). Given a $\lambda > 0$ and a kernel k defining an RKHS $(\mathcal{H}, \langle\cdot,\cdot\rangle_{\mathcal{H}})$ of functions mapping $\mathcal{X}$ to $\mathcal{Y} = \mathbb{R}$, the ridge regression solution approximates $f^\dagger$ with the minimizer of (20), where $\Omega(\|f\|) = \lambda \|f\|^2_{\mathcal{H}}$ and $\ell_{\mathcal{Y}}(f(X),Y) = \|f(X) - Y\|^2_{\mathcal{Y}^N}$. The optimal parameter $\boldsymbol{\alpha}$ is given by

$$\boldsymbol{\alpha} = \left(k(X,X) \; + \; \lambda \, I_N\right)^{-1} Y, \tag{36}$$

where $k(X,X)_{i,j} = k(x_i,x_j)$. Given a new input x, use (35) and (36) to write the solution as

$$f(x) = k(x,X)\left(k(X,X) \; + \; \lambda \, I_N\right)^{-1} Y, \tag{37}$$

where $k(x,X) = \left(k(x,x_1),\ldots,k(x,x_N)\right)$. Observe that for $\lambda = 0$, the function interpolates exactly the training data, meaning that $f(x_i) = y_i$. The ridge loss is given by the formula

$$\ell_{ridge}(f(X),Y) = \lambda Y^T [k(X,X) + \lambda I_N]^{-1} Y. \tag{38}$$

Since the function f is entirely determined by the kernel k and the penalty λ, we will write $\ell(X,Y)$ for $\ell_{ridge}(f(X),Y)$.

A.4 Operator-valued kernels

A.4.1 Introduction and main results

Operator-valued kernels are described in details in [6]. The need to introduce them arises when a supervised learning problem is considered in a functional setting: each attribute of the data is a function, and the label of each data is also a function. Similarly to scalar-valued kernels, operator-valued kernels are in one-to-one correspondence with RKHS of functions $f : \mathcal{X} \longrightarrow \mathcal{Y}$. With operator-valued kernels, the training of weights and biases of neural networks is equivalent to identifying the minimal function in a suitable RKHS defined by such a kernel. We provide a quick overview of the theory of operator-valued kernels as presented in [10, Sec 2],

Definition A.2 (**Operator-valued kernel**). We call $K : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$ an **operator-valued kernel** if
(1) K is Hermitian, i.e.

$$K(x,x') = K(x',x)^T \;\; \text{for } x,x' \in \mathcal{X}, \tag{39}$$

writing A^T for the adjoint of the operator A with respect to $\langle\cdot,\cdot\rangle_{\mathcal{Y}}$.
(2) non-negative, i.e.

$$\sum_{i,j=1}^{m} \langle y_i, K(x_i,x_j) y_j \rangle_{\mathcal{Y}} \geq 0, \;\; \text{for } (x_i,y_i) \in \mathcal{X} \times \mathcal{Y}, m \in \mathbb{N}. \tag{40}$$

We call K non-degenerate if $\sum_{i,j=1}^m \langle y_i, K(x_i,x_j) y_j \rangle_{\mathcal{Y}} = 0$ implies $y_i = 0$ for all i whenever $x_i \neq x_j$ for $i \neq j$.

For operator-valued kernels a feature map and space are defined as follows.

Definition A.3 (**Feature space/map**). $\mathcal{F}$ and $\psi : \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y},\mathcal{F})$ are a **feature space** and **feature map** for the kernel K if, for all $(x,x',y,y') \in \mathcal{X}^2 \times \mathcal{Y}^2$,

$$y^T K(x,x') y' = \langle \psi(x) y, \psi(x') y' \rangle_{\mathcal{F}}. \tag{41}$$

Write $\psi^T(x) \in \mathcal{L}(\mathcal{F},\mathcal{Y})$ for the adjoint of $\psi(x)$ and $\alpha^T \alpha' := \langle \alpha, \alpha' \rangle_{\mathcal{F}}$ for the inner product in $\mathcal{F}$ so that we can write $K(x,x')$ as

$$K(x,x') = \psi^T(x)\psi(x'). \tag{42}$$

For $\alpha \in \mathcal{F}$, write $\psi^T \alpha$ for the function $\mathcal{X} \longrightarrow \mathcal{Y}$ mapping x to the element y such that

$$\langle y', y \rangle_{\mathcal{Y}} = \langle y', \psi^T(x)\alpha \rangle_{\mathcal{Y}} = \langle \psi(x) y', \alpha \rangle_{\mathcal{F}}. \tag{43}$$

The following theorem characterizes the RKHS of an operator-valued kernel.

Theorem A.2 (**RKHS of an operator-valued kernel**). *The RKHS $\mathcal{H}$ defined by the kernel (42) is*

$$\mathcal{H} = span\{\psi^T \alpha \mid \alpha \in \mathcal{F} : \|\alpha\|_{\mathcal{F}}^2 < \infty\}, \tag{44}$$

which corresponds to the linear span of $\psi^T \alpha$ over $\alpha \in \mathcal{F}$ with finite norm. Furthermore,

$$\langle \psi^T(\cdot)\alpha, \psi^T(\cdot)\alpha' \rangle_{\mathcal{H}} = \langle \alpha, \alpha' \rangle_{\mathcal{F}} \quad and \quad \|\psi^T(\cdot)\alpha\|_{\mathcal{H}}^2 = \|\alpha\|_{\mathcal{F}}^2 \tag{45}$$

for $\alpha, \alpha' \in \mathcal{F}$.

The RKHS of an operator-valued kernel $K : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$ is a Hilbert space of functions $f : \mathcal{X} \longrightarrow \mathcal{Y}$. This allows us to extend kernel learning to problems where the output belongs to a set $\mathcal{Y} \neq \mathbb{R}$. Similarly to scalar valued kernels, the Representer theorem also holds in the case of operator-valued kernels (see [6, Appendix B] for a proof) and is of the form

$$F(\cdot) = \sum_{i=1}^{N} K(x_i, \cdot) u_i(\cdot),$$

where the weights u_i are $\mathcal{Y}$—valued functions.

A.4.2 Scalar operator-valued kernels

Scalar operator-valued (SOV) kernels correspond to operator-valued kernels defined by scalar kernels.

Definition A.4 (**Scalar operator-valued kernel**). An operator-valued kernel $K : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$ is **scalar** if

$$K(x, x') = k(x, x') I_{\mathcal{Y}}, \tag{46}$$

for some scalar-valued kernel $k : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$, and we have

$$\langle y, K(x, x') y' \rangle_{\mathcal{Y}} = k(x, x') \langle y, y' \rangle_{\mathcal{Y}}. \tag{47}$$

Given a feature space $\mathfrak{F}$ and feature map $\varphi : \mathcal{X} \longrightarrow \mathfrak{F}$ of a scalar-valued kernel k, we can construct a feature space/map for the SOV kernel $K(x, x') = k(x, x') I_{\mathcal{Y}}$. For $\beta \in \mathfrak{F}$ and $y \in \mathcal{Y}$ write $y\beta^T \in \mathcal{L}(\mathfrak{F}, \mathcal{Y})$ for the outer product between y and β defined as the linear function mapping

$$\begin{aligned} y\beta^T : \mathfrak{F} &\longrightarrow \mathcal{Y} \\ \beta' &\longmapsto y \langle \beta, \beta' \rangle_{\mathfrak{F}}. \end{aligned} \tag{48}$$

We now have all the tools to define a feature space of K.

Theorem A.3 (**Feature space of scalar operator-valued kernels**). *A feature space of the operator-valued kernel K is $\mathcal{F} := \mathcal{L}(\mathfrak{F}, \mathcal{Y})$ and its corresponding feature map $\psi : \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y}, \mathcal{F})$ is defined by*

$$\psi(x) y = y(\varphi(x))^T \ \text{ for } (x, y) \in \mathcal{X} \times \mathcal{Y}. \tag{49}$$

Furthermore,

$$\psi^T(x)\alpha = \alpha \varphi(x) \ \text{ for } x \in \mathcal{X} \ \text{ and } \ \alpha \in \mathcal{F}. \tag{50}$$

Writing $\mathcal{H}$ for the RKHS (44) defined by K,

$$\|\psi^T(\cdot)\alpha\|_{\mathcal{H}}^2 = \|\alpha\|_{\mathcal{F}}^2 = \|\alpha\|_{\mathcal{L}(\mathfrak{F}, \mathcal{Y})}^2 = Tr[\alpha^T \alpha], \tag{51}$$

where Tr is the Trace operator.

In [10], there is no mention of the finite-dimensional case $\dim \mathfrak{F} < \infty$, so we formulate it in the next proposition.

Lemma A.4. *Let $\mathcal{X} = \mathbb{R}^p$, $\mathcal{Y} = \mathbb{R}^d$ and $K(x, x') = k(x, x') I_{\mathcal{Y}}$ for a scalar kernel k with finite feature space $\mathfrak{F} = \mathbb{R}^D$ and feature map $\varphi : \mathcal{X} \longrightarrow \mathfrak{F}$. Then, the RKHS of K is*

$$\mathcal{H} = span\left\{\alpha\, \varphi(\cdot) \mid \alpha \in \mathbb{R}^{d \times D}\right\} \tag{52}$$

where the set of linear operators $\alpha \in \mathcal{L}(\mathbb{R}^D, \mathbb{R}^d)$ is identified with matrices $\mathbb{R}^{d \times D}$. Additionally, (51) corresponds to the matrix Frobenius norm in $\mathbb{R}^{d \times D}$.

Proof. Follows directly from Thm A.3. □

Example A.5. If $k(x, x') = x^T x'$ is the linear kernel, then $\varphi(x) = x$ and $\mathfrak{F} = \mathbb{R}^p$. The operator-valued kernel $K(x, x') = k(x, x') I_{\mathbb{R}^d}$ has feature space $\mathcal{F} = \mathbb{R}^{p \times p}$ and its RKHS is $\mathcal{H} = \text{span}\left\{\alpha\, I_{\mathbb{R}^p} \mid \alpha \in \mathbb{R}^{p \times d}\right\} = \mathbb{R}^{p \times d}$.

A.4.3 Kernels from activation functions

Scalar-valued kernels correspond to an inner product in a Hilbert space through a feature map (see Eq. (29)). Thus, we can define them with feature maps using an activation function. This makes a connection between the architecture of neural networks (Sec. A.2) and RKHS of functions. Consider the following feature map

$$\begin{aligned} \varphi : \mathcal{X} &\longrightarrow \mathfrak{F} = \mathcal{X} \oplus \mathbb{R} \\ x &\longmapsto \begin{pmatrix} \boldsymbol{a}(x) \\ 1 \end{pmatrix} \end{aligned} \tag{53}$$

where $\boldsymbol{a} : \mathcal{X} \longrightarrow \mathcal{X}$ is an arbitrary nonlinear activation function. From φ we define the operator-valued kernel $\Gamma : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Z})$,

$$\begin{aligned} \Gamma(x, x') &= \varphi^T(x)\varphi(x')I_{\mathcal{Z}} \\ &= (\boldsymbol{a}^T(x)\boldsymbol{a}(x') + 1)I_{\mathcal{Z}}. \end{aligned} \tag{54}$$

In particular, we take $\mathcal{Z}$ to be the input space $\mathcal{X}$ or the output space $\mathcal{Y}$. The feature map ψ of Γ satisfies (49)

$$\psi(x)z = z \begin{pmatrix} \boldsymbol{a}(x) \\ 1 \end{pmatrix}^T \in \mathcal{L}(\mathcal{X} \oplus \mathbb{R}, \mathcal{Z}),$$

corresponding to the matrix operator $A : \mathcal{X} \oplus \mathbb{R} \longrightarrow \mathcal{Z}$, $A(x', y') = z\,(\boldsymbol{a}^T(x)x' + y')$. Lemma A.4 implies that the RKHS $\mathcal{H}$ consists of linear functionals $f(\cdot) = \tilde{w}\varphi(\cdot)$, where $\tilde{w} = (W, b) \in \mathcal{L}(\mathcal{X} \oplus \mathbb{R}, \mathcal{Z})$ and

$$\tilde{w}\varphi(x) = W\boldsymbol{a}(x) + b, \tag{55}$$

where W and b can be viewed as the weights and bias of a neural network layer, which are incorporated into a single variable $\tilde{w}$.

A.4.4 Random feature maps

Another way to construct scalar operator-valued kernels is through random feature maps. Kernel approximation using such maps has recently gained a lot of interest. In kernel based learning, we optimize over a parameter whose dimension corresponds to the number of data points. Thus, kernel methods are not suited to large scale machine learning. Random feature maps can overcome these limitations by reducing training and testing times of kernel based learning algorithms. In their paper [11], Ali Rahimi and Ben Recht suggest to approximate $k(x, x')$ with an inner product in a low-dimensional space using a randomized map $\boldsymbol{z} : \mathbb{R}^p \longrightarrow \mathbb{R}^R$,

$$k(x, y) = \langle \psi(x), \psi(y) \rangle_{\mathcal{H}} \approx \boldsymbol{z}(x)^T \boldsymbol{z}(y), \tag{56}$$

where ψ and $\mathcal{H}$ are the feature map and space of the kernel k. For example, the Gaussian RBF [14, p.45] is approximated using random features sampled from the probability distribution of a Fourier transform.

Owhadi suggests another approach in [10, Sec. 6]: map the input space $\mathcal{X} = \mathbb{R}^p$ to a finite-dimensional space $\mathfrak{F} = \mathbb{R}^D$ using an affine function with a randomly chosen weight and bias, then pass it through an activation function. The obtained feature map holds the form

$$\varphi(x) = \boldsymbol{a}(Wx + b), \tag{57}$$

where all the entries of W, b are independent and selected as

$$\begin{aligned} W_{i,j} &\sim \left(\frac{1}{\sqrt{p}}\right)\mathcal{N}(0, 1), \\ b_i &\sim 0.1\mathcal{N}(0, 1). \end{aligned}$$

The operator-valued kernel is then obtained as usual with

$$K(x, x') = \varphi^T(x)\varphi(x')I_{\mathcal{Y}},$$

and a function in the RKHS of $K : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{Z}$ is

$$f(x) = w\,\boldsymbol{a}(Wx + b),\ w \in \mathbb{R}^{\dim(\mathcal{Z}) \times D}.$$

The form of the feature map corresponds to the standard architecture of a multi-layer perceptron in neural networks. Such a feature map can up-sample (or down-sample) the input space into a new space $\mathbb{R}^D$. We will use such feature maps in our simulations (cf., Section 3).

B Kernel learning: a dynamical system perspective

In this section, we introduce the main results behind the dynamics of the inputs across the layers of a neural network. We present both the discrete and continuous dynamical system and discuss a method to simulate it. All the results are taken from Owhadi's paper [10] with added clarification when needed.

B.1 Discrete dynamics

B.1.1 Mechanical regression

In his paper [10], Owhadi provides a model architecture, which he calls ***mechanical regression***, that uses two RKHS of functions.

Problem B.1 (**Mechanical regression**). Let $\Gamma : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{X})$ and $K : \mathcal{X} \times \mathcal{X} \longrightarrow \mathcal{L}(\mathcal{Y})$ be two kernels defining, respectively,

- an RKHS $\mathcal{V}$ of functions mapping $\mathcal{X}$ to $\mathcal{X}$.
- an RKHS $\mathcal{H}$ of functions mapping $\mathcal{X}$ to $\mathcal{Y}$.

Consider the problem in which $f^\dagger$ is approximated by

$$f^\ddagger = f \circ \phi_L, \ \textit{where} \ \phi_L = (I + v_L) \circ \cdots \circ (I + v_1). \tag{58}$$

The functions satisfy $v_1, \ldots, v_L \in \mathcal{V}$, $f \in \mathcal{H}$ and I is the identity map. We define the optimal solution $(v_1, \ldots, v_L, f)$ as the minimizer of

$$\frac{\nu L}{2} \sum_{s=1}^{L} \|v_s\|_{\mathcal{V}}^2 \ + \ \lambda \|f\|_{\mathcal{H}}^2 \ + \ \|f(\phi_L(X)) - Y\|_{\mathcal{Y}^N}^2, \tag{59}$$

where ν and λ are positive parameters.

The transformation ϕ_L deforms the input space and is obtained from the composition of L displacements $v_s : \mathcal{X} \longrightarrow \mathcal{X}$. The function f maps the deformed inputs $\phi_L(X)$ to the output Y with the solution of a kernel ridge regression (37). The associated ridge loss is

$$\begin{aligned} \ell(\phi_L(X), Y) &\overset{(38)}{=} \lambda Y^T [K(\phi_L(X), \phi_L(X)) + \lambda I_N]^{-1} Y \\ &= \ \lambda Y^T [K^\nu(X, X) + \lambda I_N]^{-1} Y, \end{aligned} \tag{60}$$

where $K^\nu(X, X) \coloneqq K(\phi_L(X), \phi_L(X))$ is a *warped kernel*, i.e. a kernel obtained from the deformation of the inputs via the mapping ϕ_L. In the loss (59), ν penalizes deformations of X induced by mappings with large RKHS norm, whereas λ prevents overfitting of the training data (X, Y).

The deformations $(I + v_s)$ in ϕ_L can be interpreted as jumps occurring at discrete time steps. Divide the interval $[0, 1]$ into L sub-intervals $[s/L, (s+1)/L]$ for $s = 0, \ldots, L-1$ and define

$$q^{s+1} \coloneqq (I + v_s) \circ \cdots \circ (I + v_1)(X) \tag{61}$$

for $1 \le s \le L$ and $q^1 = X$. Then, define the position $q(t)$ of the inputs as follows

- For $t = t_s \coloneqq s/L, \quad q(t) = q^{s+1}$.
- For $t_s \le t < t_{s+1}, \quad q(t) = q(t_s)$.

In particular, $q(1) = q^{L+1} = \phi_L(X)$. Thus, $q^1, \ldots, q^{L+1}$ describes the discrete trajectory $q(t)$ of the input variables as it gets deformed by the function ϕ_L over $t \in [0, 1]$. As we increase the number of layers L, we make jumps at smaller intervals of time, which lead to a continuous trajectory as $L \to \infty$. In this section, we characterize the behaviour of the discrete ($L < \infty$) and continuous ($L \to \infty$) trajectory as presented in [10].

B.1.2 Discrete least action principle

For fixed λ and ν, f is completely determined by the warped kernel K^ν, so we only need to optimize over the functions $v_1, \ldots, v_L$ which define this kernel. By defining $q^1, \ldots, q^{L+1}$ with (61), we minimize over a discrete trajectory in the input space $\mathcal{X}^N$ instead of minimizing over mappings in the RKHS $\mathcal{V}$. This is the statement of the next theorem.

Theorem B.1 (**Discrete dynamic of minimizers**). *$v_1, \ldots, v_L \in \mathcal{V}$ is a minimizer of (59) if and only if*

$$v_s(x) = \Gamma(x, q^s)\, \Gamma(q^s, q^s)^{-1} (q^{s+1} - q^s) \ \textit{for} \ x \in \mathcal{X}, \ s \in \{1, \ldots, L\}, \tag{62}$$

where $q^1, \ldots, q^N \in \mathcal{X}^N$ are defined in (61) and minimize the discrete least action principle (cf., Appendix C)

$$\begin{cases} \textit{Minimize} & \frac{\nu}{2} \sum_{s=1}^{L} \left(\frac{q^{s+1} - q^s}{\Delta t} \right)^T \Gamma(q^s, q^s)^{-1} \left(\frac{q^{s+1} - q^s}{\Delta t} \right) \Delta t \ + \ \ell(q^{L+1}, Y), \\ \textit{over} & q^2, \ldots, q^{L+1} \in \mathcal{X}^N \ \textit{with} \ q^1 = X, \end{cases} \tag{63}$$

where $\Delta t \coloneqq 1/L$. Introducing the momentum variable

$$p^s \coloneqq \Gamma(q^s, q^s)^{-1} \frac{q^{s+1} - q^s}{\Delta t}, \tag{64}$$

(q^s, p^s) follows the discrete Hamiltonian dynamics

$$\begin{cases} q^{s+1} = q^s + \Delta t \Gamma(q^s, q^s) p^s \\ p^{s+1} = p^s - \frac{\Delta t}{2} \partial_{q^{s+1}} \left((p^{s+1})^T \Gamma(q^{s+1}, q^{s+1}) p^{s+1} \right), \end{cases} \tag{65}$$

with $q^1 = X$ and p^1 minimizes

$$\frac{\nu}{2} \sum_{s=1}^{L} (p^s)^T \Gamma(q^s, q^s) p^s \, \Delta t \ + \ \ell(q^{L+1}, Y). \tag{66}$$

Eq. (62) is the representer theorem for operator-valued kernels obtained from the exact interpolation $v_s(q^s) = q^{s+1} - q^s$. The matrix $\Gamma(q^s, q^s)$ is a block operator matrix with entries $\Gamma(q_i^s, q_j^s)$ (see [6, Def. 2]).

Equation (63) is a discrete variational problem whose minimizer satisfies the discrete Euler-Lagrange equations. This is a classical result from discrete mechanics and we refer the reader to [7, 9] for a standard reference on this topic. The discrete Hamiltonian system (65) is obtained using the correspondence between Lagrangian and Hamiltonian mechanics, where the Lagrangian is

$$\mathfrak{L}(q, \dot{q}) := \frac{1}{2} \dot{q}^T \Gamma(q, q)^{-1} \dot{q}, \tag{67}$$

and the loss in (63) approximates

$$\nu \sum_{s=1}^{L} \mathfrak{L}(q_{s/L}, \dot{q}_{s/L}) \Delta t \; + \; \ell(q^{L+1}, Y). \tag{68}$$

B.2 Continuous dynamics

B.2.1 Idea registration: infinite-depth limit

The solution of mechanical regression is a function represented by a neural network with L hidden layers. The infinite-depth limit $L \to \infty$ is called *idea registration*. The next theorem characterizes the solution of this problem.

Theorem B.2 (Idea registration). *As $L \to \infty$, the minimizer $f \circ \phi_L$ of Pb. (B.1) converges to a minimizer of the form $f \circ \phi^v(X, 1)$ where (v, f) are minimizers of*

$$\min_{f, v} \frac{\nu}{2} \int_0^1 \|v(\cdot, t)\|_{\mathcal{V}}^2 \, dt \; + \; \lambda \|f\|_{\mathcal{H}}^2 \; + \; \|f(\phi^v(X, 1)) - Y\|_{\mathcal{Y}^N}^2, \tag{69}$$

and $\phi^v(x, t)$ is the flow map of v defined as the solution of

$$\begin{cases} \dot{\phi}(x, t) = v(\phi(x, t), t) & \text{for } (x, t) \in \mathcal{X} \times [0, 1], \\ \phi(x, 0) = x & \text{for } x \in \mathcal{X}, \end{cases} \tag{70}$$

where v belongs to the space $C([0, 1], \mathcal{V})$ of continuous functions $v : \mathcal{X} \times [0, 1] \longrightarrow \mathcal{X}$ such that

1. *$x \longmapsto v(x, t)$ is a map from $\mathcal{X}$ to $\mathcal{V}$ for all $t \in [0, 1]$,*
2. *$v(x, t)$ is uniformly Lipschitz continuous (in t and x).*

The optimal deformation of the inputs is defined by the flow map $\phi^v(\cdot, t)$. Let $q_i(t) := \phi^v(x_i, t)$, then each input propagates according to the ODE

$$\begin{cases} \dot{q}_i(t) = v(q_i(t), t), & \text{for } t \in [0, 1] \\ q_i(0) = x_i, \end{cases} \tag{71}$$

which is completely determined by v.

We explain how to get (69), (70) from mechanical regression with a formal derivation (cf., [10] for a proof). Consider $v_1, \ldots, v_L$ in (58) and introduce $\Delta t = 1/L$, $v_s = \Delta t \, \tilde{v}_s$. Then, $\{\tilde{v}_s\}_{s=1}^L$ minimize

$$\frac{\nu}{2} \sum_{s=1}^{L} \|\tilde{v}_s\|_{\mathcal{V}}^2 \, \Delta t \; + \; \ell(\phi_L(X), Y). \tag{72}$$

The discrete dynamic of the flow map is

$$\frac{\phi_{s+1}(x) - \phi_s(x)}{\Delta t} = \tilde{v}_{s+1}(\phi_s(x)). \tag{73}$$

As $L \to \infty$, $\phi_s = (I + \Delta t \, \tilde{v}_s) \circ \cdots \circ (I + \Delta t \, \tilde{v}_1)$ and $\tilde{v}_s$ approximate ϕ^v and v at time $t_s := \frac{s}{L}$. The sum in (72) becomes the integral

$$\frac{\nu}{2} \int_0^1 \|v(\cdot, t)\|_{\mathcal{V}}^2 \, dt$$

and the flow map satisfies the continuous dynamic (70).

B.2.2 Continuous least action principle

By considering a continuous trajectory $q \in C^1([0,1], \mathcal{X}^N)$, the sum in (68) becomes the action integral

$$\mathcal{A}[q] = \int_0^1 \mathfrak{L}(q(t), \dot{q}(t))\, dt. \tag{74}$$

Hence, a continuous trajectory evolves according to the following least action principle

$$\begin{cases} \text{Minimize} & v\,\mathcal{A}[q] \;+\; \ell(q(1), Y) \\ \text{over} & q \in C^1([0,1], \mathcal{X}^N), \text{ subject to } q(0) = X. \end{cases} \tag{75}$$

Similarly to mechanical regression, the minimizer v of (69) can be reduced to the minimizer of the continuous least action principle (75).

Theorem B.3 (**Continuous dynamics of minimizers**). *Let (f, v) be the minimizers of (69), then*

$$v(x,t) = \Gamma(x,q)\Gamma(q,q)^{-1}\dot{q}, \tag{76}$$

where $q(t)$ is the minimizer of the least action principle (75).

Let p be the momentum variable defined as $p := \Gamma(q,q)^{-1}\dot{q}$. Therefore, the flow map is of the form

$$\begin{cases} \dot{\phi}^v(x,t) = \Gamma(\phi^v(x,t), q)p & \text{for } (x,t) \in \mathcal{X} \times [0,1] \\ \phi^v(x,0) = x & \text{for } x \in \mathcal{X}. \end{cases} \tag{77}$$

The following result characterizes the Hamiltonian dynamics of the trajectory in idea registration.

Corollary B.4 (**Hamiltonian system**). *Let $q \in \mathcal{X}^N$ starting at $q(0) = X$ be the minimizer of (75). Then, the pair of position and momentum variables (q,p), with $p := \Gamma(q,q)^{-1}\dot{q}$, is a solution of the Hamiltonian system*

$$\begin{cases} \dot{q}_i = \partial_{p_i} \mathfrak{H}(q,p) \\ \dot{p}_i = -\partial_{q_i} \mathfrak{H}(q,p), \end{cases} \tag{78}$$

where the Hamiltonian is

$$\begin{aligned} \mathfrak{H}(q,p) &= \frac{1}{2} p^T \Gamma(q,q) p \\ &= \frac{1}{2} \sum_{i,j=1}^N p_i^T \Gamma(q_i, q_j) p_j, \end{aligned} \tag{79}$$

and $p(0)$ minimizes

$$\mathfrak{B}(p(0), X, Y) := v\mathfrak{H}(X, p(0)) \;+\; \ell(q(1), Y). \tag{80}$$

Furthermore, $p(1)$ satisfies

$$v p(1) \;+\; \partial_{q(1)} \ell(q(1), Y) = 0. \tag{81}$$

The dynamics of q is completely determined by the initial momentum $p(0)$. Hence, both v and f are determined by $p(0)$. The Hamiltonian representation of minimizers of (75) reduces the search for the optimal trajectory to the search for an initial momentum $p(0)$ identified as the minimizer of (80). In image registration [1] this method is known as *geodesic shooting*.

B.2.3 Energy preservation

Eq. (76) formulates v as a function of (q,p) using the Representer theorem. A consequence of this representation is that $\|v(\cdot,t)\|_{\mathcal{V}}^2$ is constant over $t \in [0,1]$. To see this, use (33)

$$\begin{aligned} \|v(\cdot,t)\|_{\mathcal{V}}^2 &= \sum_{i,j=1}^N p_i^T \Gamma(q_i, q_j) p_j \\ &= 2\mathfrak{H}(q,p). \end{aligned}$$

Since the Hamiltonian is a constant of motion, the norm is constant in time.

B.3 Existence of minimizers

The existence and uniqueness of minimizers are discussed in details in [10, Sec. 3]. We provide the main condition on the kernel Γ to guarantee existence of minimizers of mechanical regression and idea registration.

Lemma B.5. *Assume that*

(1) there exists $r > 0$ such that $Z^T \Gamma(X,X) Z > r Z^T Z$ for all $Z \in \mathcal{X}^N$,

(2) Γ admits $\mathcal{F}$ and ψ as feature space/map, $\mathcal{F}$ is finite-dimensional, ψ and its first and second order partial derivatives are continuous and uniformly bounded,

(3) $\mathcal{X}$ is finite-dimensional.

(1) is equivalent to the non singularity of $\Gamma(X,X)$ and (2) implies that $(x,x') \to \Gamma(x,x')$ and its first and second order partial derivatives are continuous and uniformly bounded.

B.4 Trajectory optimization: geodesic shooting

The formulation of mechanical regression with RKHS spaces $\mathcal{V}, \mathcal{H}$ in Problem (B.1) is a minimization problem over functions in $\mathcal{V}$, which defines a trajectory satisfying the least action principle (75). In Sec. B.2.2 we characterized the optimal trajectory of idea registration using the initial momentum $p(0)$ (Corollary B.4). To minimize the loss $\mathfrak{B}(p(0), X, Y)$ given by (80), we need to obtain $q(1)$ as a function of $p(0)$, which requires us to solve the Hamiltonian system (78). In general we need to discretize the system with a suitable integrator.

B.4.1 $p(0)$ optimization

1. Set $p(0) = 0$ which corresponds to no deformation of the inputs, i.e. $q(1) = X$.
2. Discretize the Hamiltonian (78) with L discrete time steps using a suitable integrator to obtain a discrete trajectory where $q^{L+1} = q(1)$.
3. Calculate the gradient of $\mathfrak{B}(p(0), X, Y)$ with respect to $p(0)$.
4. Update $p(0)$ using gradient descent
$$p(0) \leftarrow p(0) - \gamma \partial_{p(0)} \mathfrak{B}$$
or any other gradient based method.
5. Repeat (2)-(4) for the desired number of steps or until convergence of $\mathfrak{B}(p(0), X, Y)$.

Optimizing $p(0)$ is a $N \times \dim(\mathcal{X})$ dimensional minimization problem. Thus, it does not scale well as we increase the number of training points. Additionally, it requires to simulate a mechanical system using a discrete integrator. Although the discrete dynamics of mechanical regression (65) is a natural choice, this integrator is implicit, which makes it a poor choice from a computational point of view. We will prefer the use of an explicit and non-separable integrator. See [8] for a discussion on variational integrators.

C Least action principle

We start this section by introducing the ***least action principle***, also known as the ***principle of critical action***, which appears in classical mechanics. We provide a brief overview of some concepts from Lagrangian and Hamiltonian mechanics taking the notation from [9] and [8] (with a slight modification).

In Lagrangian mechanics, the framework is as follows:

- Identify a ***configuration space*** Q with coordinates $(q_1, \dots, q_n)$.
- Form the velocity phase space TQ, also called the ***tangent bundle*** of Q. Coordinates on TQ are denoted by $(q_1, \dots, q_n, \dot{q}_1, \dots, \dot{q}_n)$.
- The ***Lagrangian*** is regarded as a function $\mathfrak{L} : TQ \longrightarrow \mathbb{R}$. We use the shorthand notation $\mathfrak{L}(q(t), \dot{q}(t))$ or simply $\mathfrak{L}(q, \dot{q})$ when referring to $\mathfrak{L}(q_1(t), \dots, q_n(t), \dot{q}_1(t), \dots, \dot{q}_n(t))$. The coordinate notation $\mathfrak{L}(q_i(t), \dot{q}_i(t))$ will be useful for proving the results in this section.

We can now define the ***action*** of a curve $q(t)$ joining two fixed points in Q over a time interval $[t_0, t_1]$ as follows,

$$\mathcal{A}[q] = \int_{t_0}^{t_1} \mathfrak{L}(q(t), \dot{q}(t))\, dt. \tag{82}$$

The action functional depends on t_0 and t_1, but this is not explicit in the notation. The idea behind Hamilton's principle is to find the curve $q(t)$ for which the functional $\mathcal{A}$ is stationary under variations of $q(t)$ with fixed endpoints. By ***variation*** we mean a function δq, such that $q + \delta q$ is a small modification of q with same endpoints. We think of it as a family of curves with respect to a parameter ϵ. We illustrate this idea with the example below.

Example C.1 (**Variation**)**.** For a fixed curve $q(t)$, consider a modification of the form

$$q^{\epsilon}(t) = q(t) + \epsilon\, \eta(t),$$

where $\eta(t)$ is a smooth curve $\eta(t_0) = \eta(t_1) = 0$. Then, we write $\delta q = \eta$.

More generally, for a family of smooth parametric curves q^{ϵ}, corresponding to small modifications of q, we write

$$\left.\frac{dq^{\epsilon}}{d\epsilon}\right|_{\epsilon=0} = \delta q. \tag{83}$$

We can now state the least action principle [9, p.221].

Definition C.1 (**Path space**)**.** Let Q be a manifold and let $\mathfrak{L} : TQ \longrightarrow \mathbb{R}$ be a **regular Lagrangian** (def. in [9, Sect. 7.2, p.183]). Fix two points y_0 and y_1 in Q and an interval $[t_0, t_1]$, and define the **path space** from y_0 to y_1 by

$$\Omega(y_0, y_1, [t_0, t_1]) = \{q : [t_0, t_1] \longrightarrow Q \,|\, q \text{ is a } C^2 \text{ curve},\ q(t_0) = y_0,\ q(t_1) = y_1\}. \tag{84}$$

Theorem C.1 (**Variational principle of Hamilton**). *Let $\mathfrak{L}$ be a Lagrangian on TQ. A curve $q : [t_0, t_1] \longrightarrow Q$ joining $y_0 = q(t_0)$ to $y_1 = q(t_1)$ satisfies the Euler-Lagrange equations*

$$\frac{d}{dt}\left(\frac{\partial \mathfrak{L}}{\partial \dot{q}_i}\right) = \frac{\partial \mathfrak{L}}{\partial q_i} \tag{85}$$

if and only if q is a critical point of the function $\mathcal{A} : \Omega(y_0, y_1, [t_0, t_1]) \longrightarrow \mathbb{R}$, that is

$$d\mathcal{A}(q(t)) \cdot \delta q(t) := \frac{d}{d\epsilon}\bigg|_{\epsilon=0} \mathcal{A}[q_\epsilon(t)] = 0 \tag{86}$$

for all $\delta q(t)$ with $\delta q(t_0) = \delta q(t_1) = 0$, where q^ϵ is a smooth family of curves with $q_0 = q$ and with δq defined in (83).

Proof. Passing the derivative under the integral and using integration by parts, yields

$$\begin{aligned} d\mathcal{A}(q(t)) \cdot \delta q(t) &= \frac{dq^\epsilon}{d\epsilon}\bigg|_{\epsilon=0} \int_{t_0}^{t_1} \mathfrak{L}(q_i^\epsilon(t), \dot{q}_i^\epsilon(t))\, dt \\ &= \int_{t_0}^{t_1} \delta q_i \left(\frac{\partial \mathfrak{L}}{\partial q_i} - \frac{d}{dt}\frac{\partial \mathfrak{L}}{\partial \dot{q}_i}\right) dt \; + \; \frac{\partial \mathfrak{L}}{\partial \dot{q}_i}\, \delta q_i \Big|_{t_0}^{t_1}. \end{aligned} \tag{87}$$

Since (86) and the boundary conditions for δq hold, the Euler-Lagrange (85) equation must be satisfied. □

Lemma C.2. *The solution of (75) satisfies the Euler-Lagrange equation with the Lagrangian defined in (67).*

Proof. Consider the setting of Theorem C.1 with $Q = \mathcal{X}^N$, $t_0 = 0$, $t_1 = 1$ and $y_0 = X$. Let q be the minimizer of (75) and set $y_1 = q(1)$. Finally, define $S[q] := \nu \mathcal{A}[q] + \ell(q(1), Y)$.

The term $\ell(q^\epsilon(1), Y)$ is constant for all smooth curves q^ϵ satisfying the conditions of Theorem C.1. This implies that the functional derivative $\mathcal{A}[q]$ of the minimizing curve q must be equal to 0, from which the Euler-Lagrange equation follows. □

References

[1] Stéphanie Allassonnière, Alain Trouvé, and Laurent Younes. Geodesic shooting and diffeomorphic matching via textured meshes. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 365–381. Springer Berlin Heidelberg, 2005.

[2] John Cardy. *Mean field theory*, pages 16–27. Cambridge Lecture Notes in Physics. Cambridge University Press, 1996.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[4] David Harrison and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5(1):81–102, 1978.

[5] Thomas Hofmann, Bernhard Schölkopf, and Alexander J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, 36(3), Jun 2008.

[6] Hachem Kadri, Emmanuel Duflos, Philippe Preux, Stéphane Canu, Alain Rakotomamonjy, and Julien Audiffren. Operator-valued kernels for learning from functional response data. *Journal of Machine Learning Research*, 17(20):1–54, 2016.

[7] J. E. Marsden and M. West. Discrete mechanics and variational integrators. *Acta Numerica*, 10:357–514, 2001.

[8] Jerrold E. Marsden, George W. Patrick, and Steve Shkoller. Multisymplectic geometry, variational integrators, and nonlinear pdes. *Communications in Mathematical Physics*, 199:351–395, Dec 1998.

[9] Jerrold E. Marsden and Tudor S. Ratiu. *Introduction to Mechanics and Symmetry*. Springer-Verlag, New York, 2nd edition, 1999.

[10] Houman Owhadi. Do ideas have shape? Plato's theory of forms as the continuous limit of artificial neural networks. https://arxiv.org/abs/2008.03920, 2020.

[11] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

[12] Sebastian Ruder. An overview of gradient descent optimization algorithms. http://arxiv.org/abs/1609.04747, 2016.

[13] Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 515–521, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[14] Bernhard Schölkopf, Alexander J. Smola, and Francis Bach. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2001.

[15] Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer-Verlag, New York, 2008.

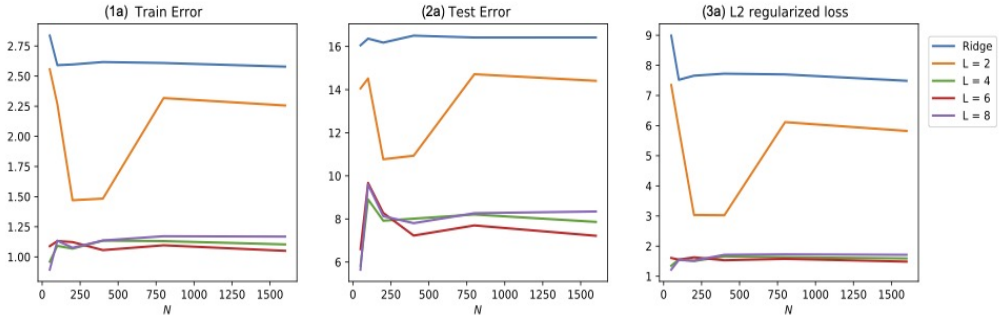

Figure 1: ReLU

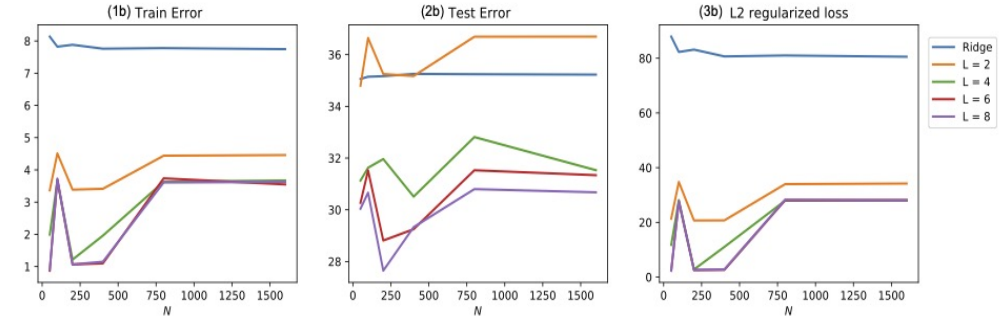

Figure 2: tanh

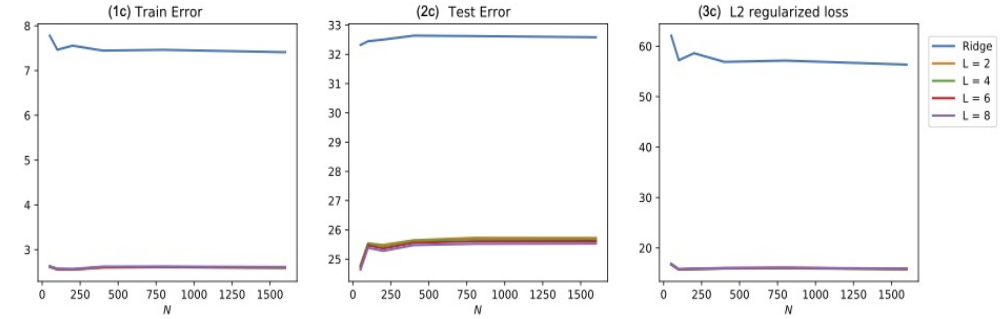

Figure 3: sigmoid

Figure 4: Ridge regression vs. Mechanical regression against N on the synthetic dataset. N corresponds to the number of training samples which are used to train α. (1) Comparison of the MSE on the training set; (2) comparison on the test set. (3) Comparison between the L_2 regularized loss (19). Ridge regression corresponds to the untrained model with $\alpha_s = 0$. Mechanical regression is obtained by minimizing (19) with respect to the parameters $\{\alpha_s\}_{s=1}^{L}$ with $L \in \{2, 4, 6, 8\}$. The $x-$axis is in log scale.

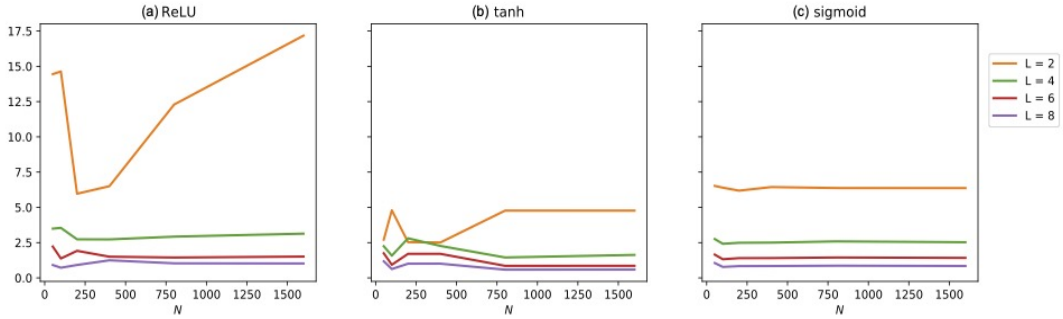

Figure 5: Energy $E(\alpha)$ of mechanical regression for $L \in \{2, 4, 6, 8\}$ against N on the synthetic dataset. Mechanical regression is obtained by minimizing (19) with respect to the parameters $\{\alpha_s\}_{s=1}^{L}$ with $L \in \{2, 4, 6, 8\}$.

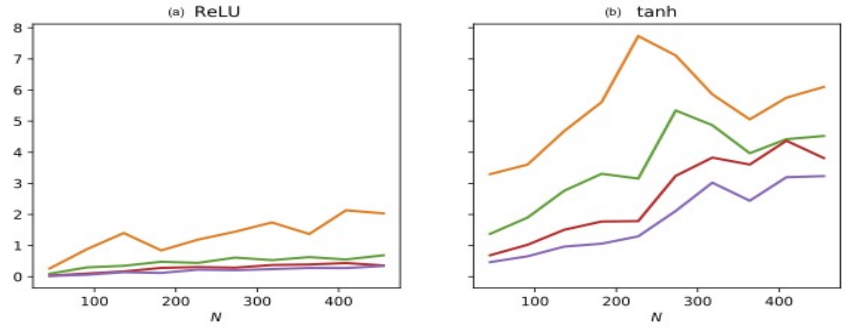

Figure 6: Energy $E(\alpha)$ of mechanical regression for $L \in \{2,4,6,8\}$ against N on the Boston housing dataset.

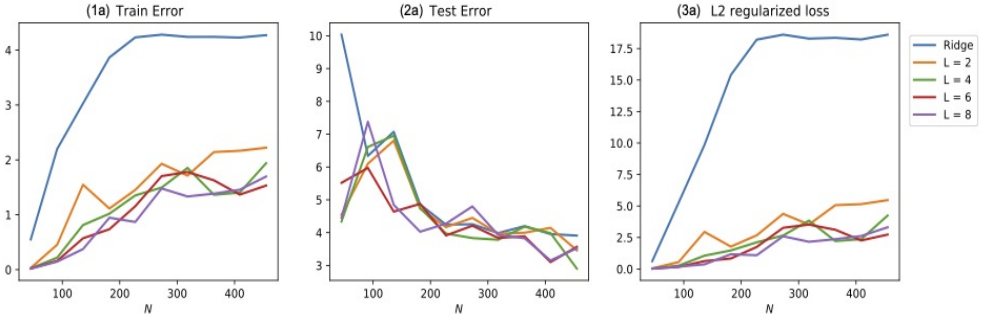

Figure 7: ReLU

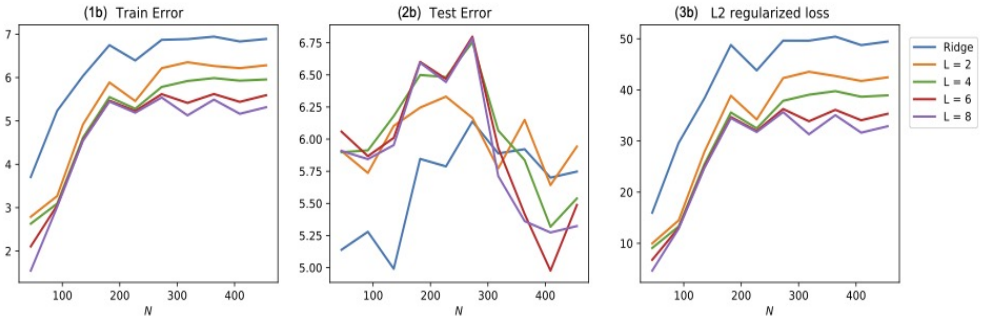

Figure 8: tanh

Figure 9: Ridge regression vs. Mechanical regression against N on the Boston housing dataset. N corresponds to the number of training samples which are used to train α. (1) Comparison of the MSE on the training set; (2) comparison on the test set. (3) Comparison between the L_2 regularized loss (19). Ridge regression corresponds to the untrained model with $\alpha_s = 0$. Mechanical regression is obtained by minimizing (19) with respect to the parameters $\{\alpha_s\}_{s=1}^{L}$ with $L \in \{2,4,6,8\}$.

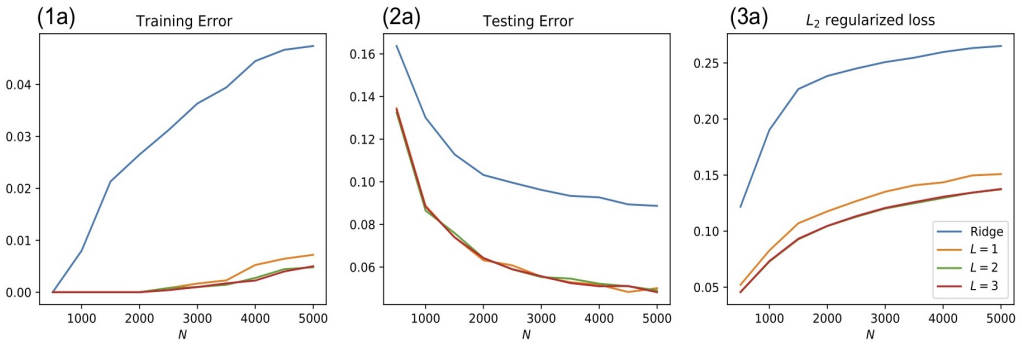

Figure 10: ReLU

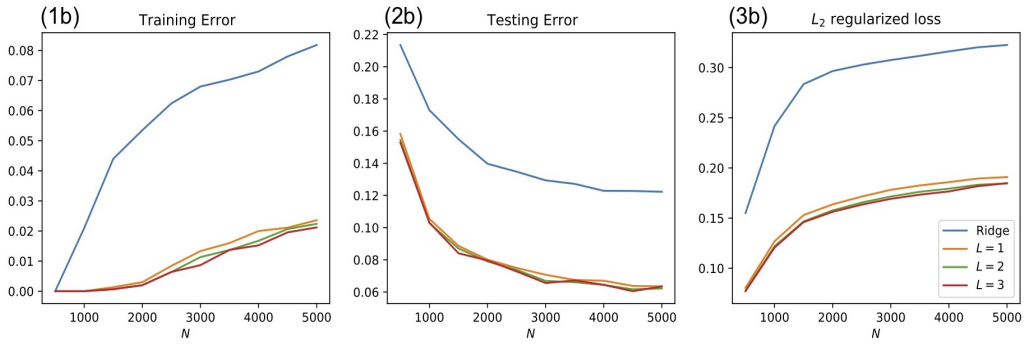

Figure 11: tanh

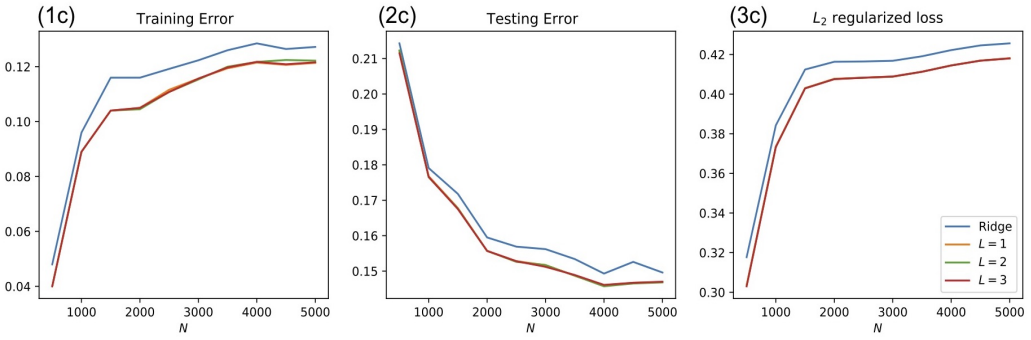

Figure 12: sigmoid

Figure 13: Ridge regression vs. Mechanical regression against N on MNIST. (1) Comparison of the error on the training set; (2) comparison on the test set. The error corresponds to the number of misclassified samples. (3) Comparison between the L_2 regularized loss (19). Ridge regression corresponds to the untrained model with $\alpha_s = 0$. Mechanical regression is obtained by minimizing (19) with respect to the parameters $\{\alpha_s\}_{s=1}^{L}$ with $L \in \{1,2,3\}$.

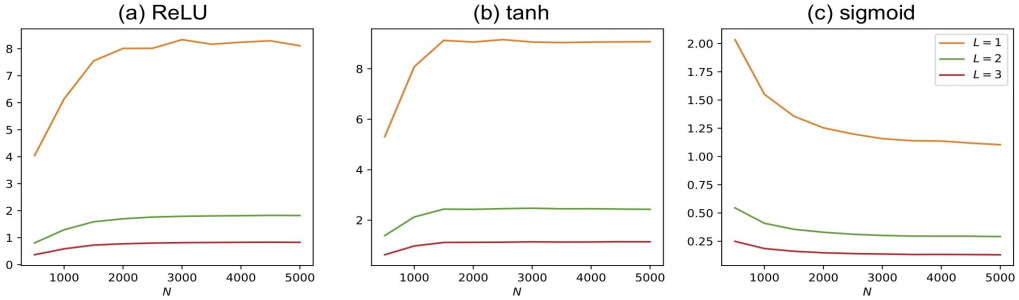

Figure 14: Energy $E(\alpha)$ of mechanical regression for $L \in \{1,2,3\}$ against N on MNIST.

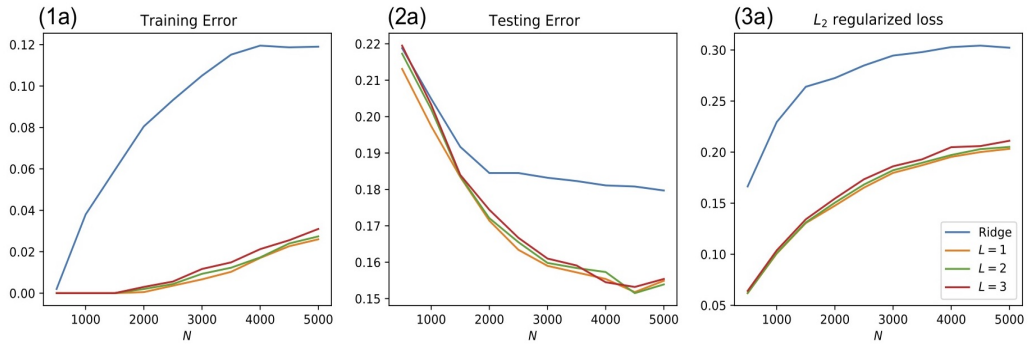

Figure 15: ReLU

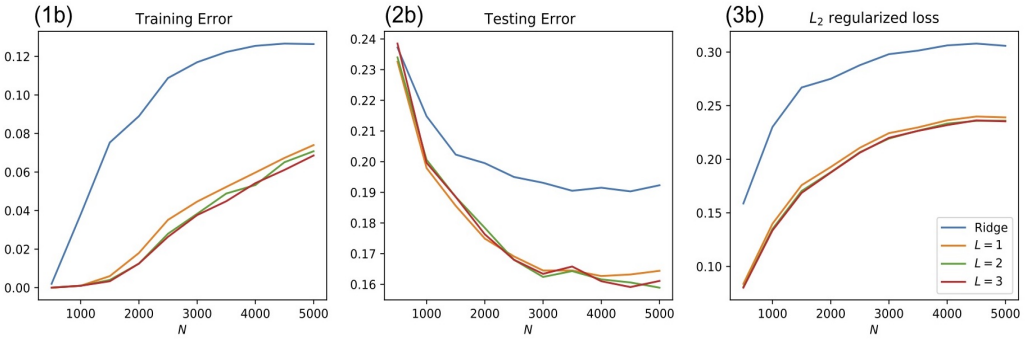

Figure 16: tanh

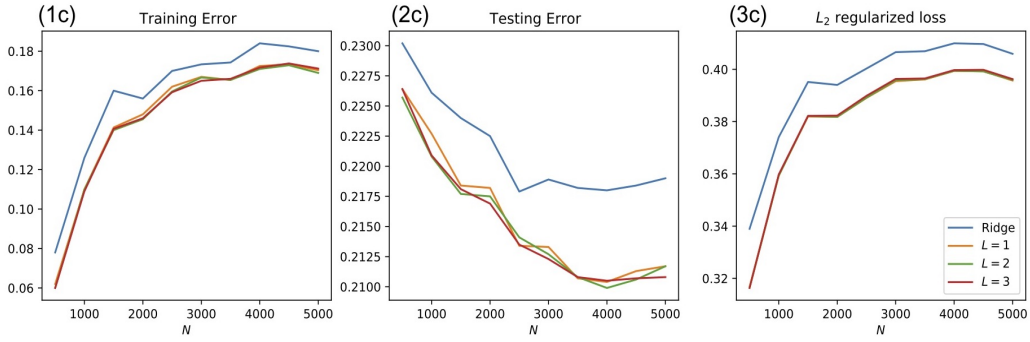

Figure 17: sigmoid

Figure 18: Ridge regression vs. Mechanical regression against N on Fashion MNIST. (1) Comparison of the error on the training set; (2) comparison on the test set. The error corresponds to the number of misclassified samples. (3) Comparison between the L_2 regularized loss (19). Ridge regression corresponds to the untrained model with $\alpha_s = 0$. Mechanical regression is obtained by minimizing (19) with respect to the parameters $\{\alpha_s\}_{s=1}^{L}$ with $L \in \{1, 2, 3\}$.

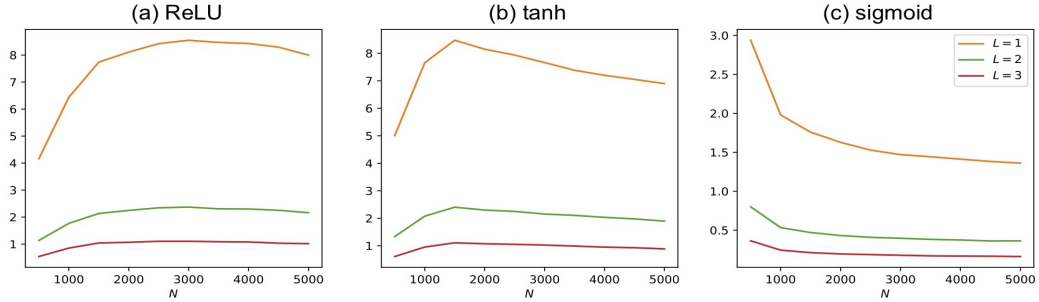

Figure 19: Energy $E(\alpha)$ of mechanical regression for $L \in \{1, 2, 3\}$ against N on Fashion MNIST.