Kernel Methods and Machine Learning

Workshop: The Mathematics of Scientific Machine Learning and Digital Twins

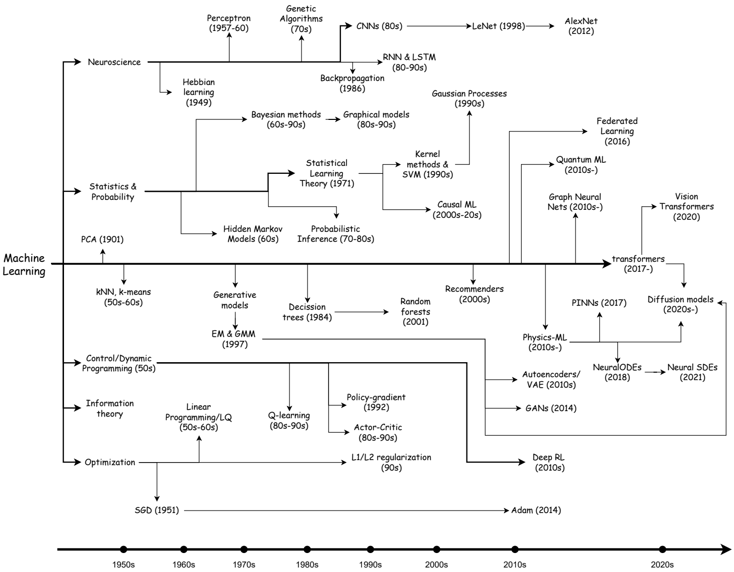
Daniel López Montero

Chair for Dynamics, Control, Machine Learning and Numerics
Friedrich-Alexander-Universität

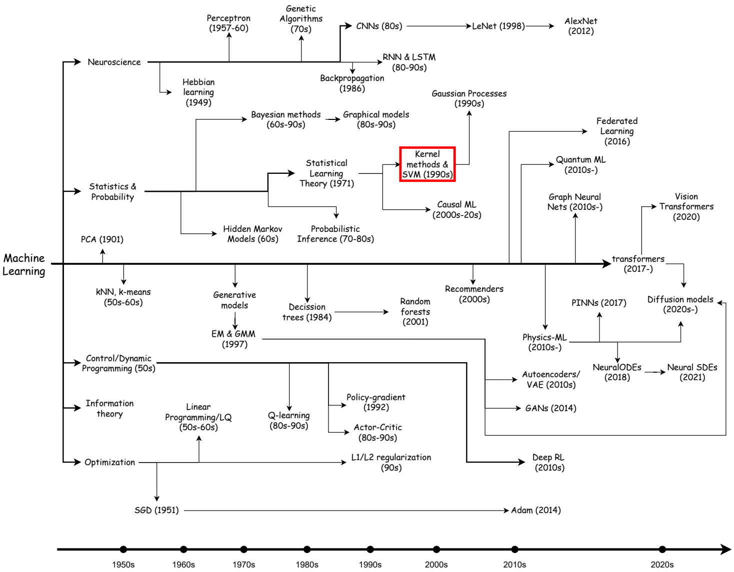
November 19-25, 2025

Overview

History of Machine Learning



History of Machine Learning



Introduction

Definition

A kernel $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be positive definite if

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j \kappa(x_i, x_j) \geq 0 \qquad c_1, \ldots, c_n \in \mathbb{R},\ x_1, \ldots, x_n \in \mathcal{X}$$

Examples of kernels:

$$\kappa(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right) \qquad \text{(Gaussian kernel)}$$

$$\kappa(x, x') = \exp\langle x, x' \rangle \qquad \text{(Exponential kernel)}$$

$$\kappa(x, x') = (\langle x, x' \rangle + c)^d \qquad \text{(Polynomial kernel)}$$

Introduction: Reproducing Kernel Hilbert Space

We want to take the closure of linear combinations of kernel functions.

$$f := \sum_{i=1}^{n} \alpha_i \kappa(\cdot, x_i), \qquad \alpha_i \in \mathbb{R}, x_i \in \mathcal{X}$$

And we take the inner product as

$$\langle f, g \rangle_{\mathcal{H}_\kappa} \equiv \left\langle \sum_{i=1}^{n} \alpha_i \kappa(\cdot, x_i), \sum_{j=1}^{m} \beta_j \kappa(\cdot, y_j) \right\rangle_{\mathcal{H}_\kappa} := \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_i \beta_j \kappa(x_i, y_j)$$

Theorem (Aronszajn 1950)

Let κ be a symmetric positive definite kernel, then there exists a unique Hilbert space $\mathcal{H}_\kappa$.

1. **Gaussian and exponential kernels:** $\mathcal{H}_\kappa \subseteq C(\mathcal{X})$ dense.

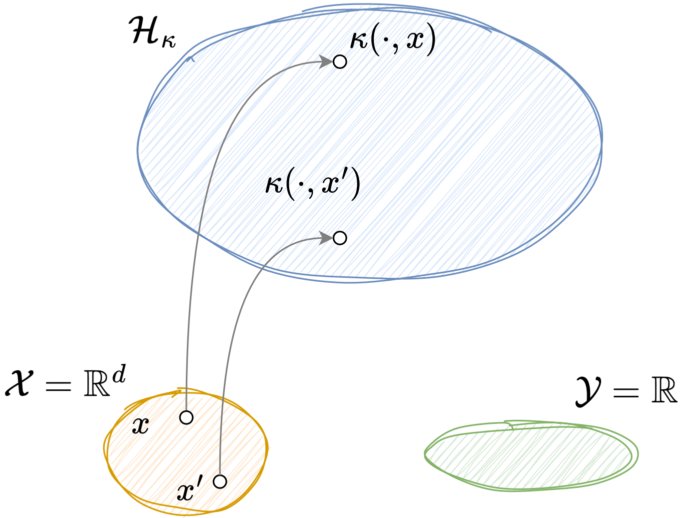
$$\kappa(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right), \qquad \kappa(x, x') = \exp\langle x, x'\rangle$$

2. **Sobolev kernels:** $\mathcal{H}_{\kappa_\nu} \subseteq H^s(\mathbb{R}^d)$ is dense for $s < \nu + d/2$. And for $s = \nu + d/2$, they coincide.

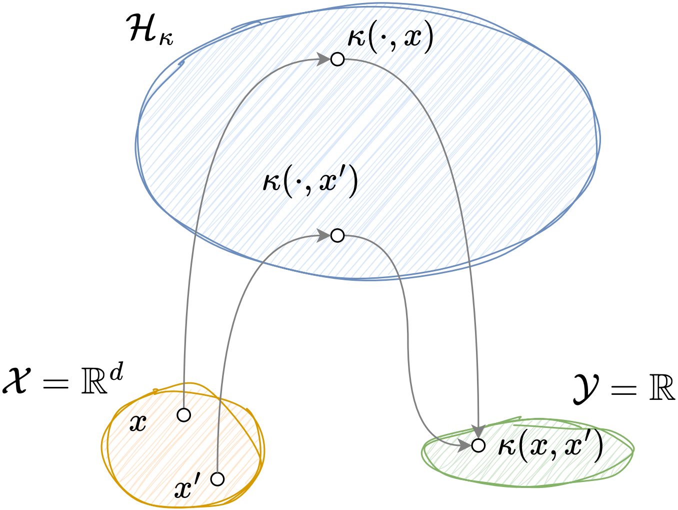
$$\kappa_\nu(x, x') = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\|x - x'\|}{\ell}\right)^\nu K_\nu\left(\frac{\|x - x'\|}{\ell}\right),$$

where K_ν is the modified Bessel function of the second kind.

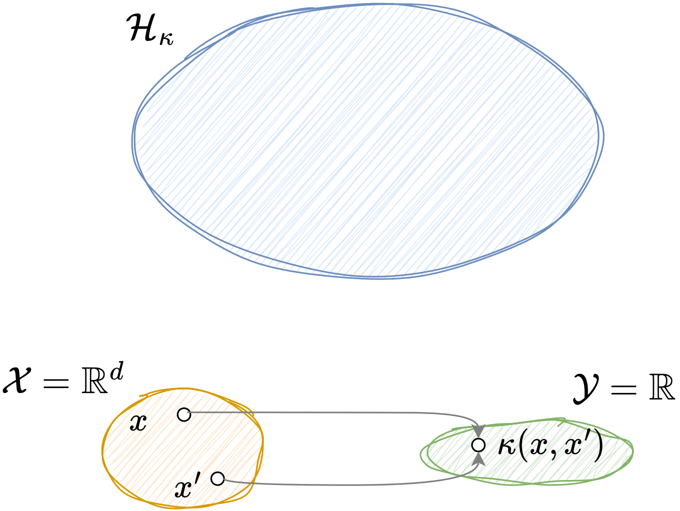
Idea of Kernel Methods



Idea of Kernel Methods



Idea of Kernel Methods



Representer Theorem

Theorem (Schölkopf, Herbrich, et al. 2001)

Suppose we are given training data:

$$(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathbb{R}.$$

We consider optimization problems of the form:

$$\min_{f \in \mathcal{H}_\kappa} L((x_1, y_1, f(x_1)), \ldots, (x_n, y_n, f(x_n))) + \lambda \|f\|^2_{\mathcal{H}_\kappa}.$$

Then, any minimizer $f^* \in \mathcal{H}_\kappa$ admits a representation of the form

$$f^*(\cdot) = \sum_{i=1}^{n} \alpha_i \kappa(\cdot, x_i)$$

for some coefficients $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$.

Self-Attention as a Kernel Method

Given a positive semi-definite matrix $\mathbf{A}$, define the kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) := \exp\{-(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)\}$$

where $\mathbf{A} = \frac{1}{2}\mathbb{Q}^\top\mathbb{K}$ and let the value projection be $\mathbf{y} = \mathbb{V}\mathbf{x}$.

Self-Attention as a Kernel Method

Given a positive semi-definite matrix $\mathbf{A}$, define the kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) := \exp\{-(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)\}$$

where $\mathbf{A} = \frac{1}{2}\mathbb{Q}^\top\mathbb{K}$ and let the value projection be $\mathbf{y} = \mathbb{V}\mathbf{x}$.
The Nadaraya-Watson smoothing kernel (Nadaraya 1964; Watson 1964):

$$\sum_j \frac{\kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_j}{\sum_k \kappa(\mathbf{x}_i, \mathbf{x}_k)}$$

Self-Attention as a Kernel Method

Given a positive semi-definite matrix $\mathbf{A}$, define the kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) := \exp\{-(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)\}$$

where $\mathbf{A} = \frac{1}{2}\mathbb{Q}^\top\mathbb{K}$ and let the value projection be $\mathbf{y} = \mathbb{V}\mathbf{x}$.
The Nadaraya-Watson smoothing kernel (Nadaraya 1964; Watson 1964):

$$\sum_j \frac{\kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_j}{\sum_k \kappa(\mathbf{x}_i, \mathbf{x}_k)} = \underbrace{\sum_j \frac{e^{\langle \mathbb{Q}\mathbf{x}_i, \mathbb{K}\mathbf{x}_j\rangle}}{\sum_k e^{\langle \mathbb{Q}\mathbf{x}_i, \mathbb{K}\mathbf{x}_k\rangle}} \mathbb{V}\mathbf{x}_j}$$

Self-Attention as a Kernel Method

Given a positive semi-definite matrix $\mathbf{A}$, define the kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) := \exp\{-(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j)\}$$

where $\mathbf{A} = \frac{1}{2}\mathbb{Q}^\top\mathbb{K}$ and let the value projection be $\mathbf{y} = \mathbb{V}\mathbf{x}$.
The Nadaraya-Watson smoothing kernel (Nadaraya 1964; Watson 1964):

$$\sum_j \frac{\kappa(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_j}{\sum_k \kappa(\mathbf{x}_i, \mathbf{x}_k)} = \underbrace{\sum_j \frac{e^{\langle \mathbb{Q}\mathbf{x}_i, \mathbb{K}\mathbf{x}_j \rangle}}{\sum_k e^{\langle \mathbb{Q}\mathbf{x}_i, \mathbb{K}\mathbf{x}_k \rangle}} \mathbb{V}\mathbf{x}_j}_{=\mathrm{Attn}(\mathbf{x}_i, X)}$$

Key, query, and value matrices are given by $\mathbb{K}, \mathbb{Q}, \mathbb{V}$ respectively.

Proof that Self-Attention is Kernel Regression

Proof.

Expand the exponent:

$$-(\mathbf{x}_i - \mathbf{x}_j)^\top \mathbf{A}(\mathbf{x}_i - \mathbf{x}_j) = -\mathbf{x}_i^\top \mathbf{A}\mathbf{x}_i - \mathbf{x}_j^\top \mathbf{A}\mathbf{x}_j + 2\mathbf{x}_i^\top \mathbf{A}\mathbf{x}_j$$

Then,

$$\begin{aligned}\frac{\kappa(\mathbf{x}_i, \mathbf{x}_j)}{\sum_k \kappa(\mathbf{x}_i, \mathbf{x}_k)} &= \frac{\exp\{-\mathbf{x}_j^\top \mathbf{A}\mathbf{x}_j + 2\mathbf{x}_i^\top \mathbf{A}\mathbf{x}_j\}}{\sum_k \exp\{-\mathbf{x}_k^\top \mathbf{A}\mathbf{x}_k + 2\mathbf{x}_i^\top \mathbf{A}\mathbf{x}_k\}} \\ &= \frac{e^{\langle \mathbb{Q}\mathbf{x}_i, \mathbb{K}\mathbf{x}_j \rangle}}{\sum_k e^{\langle \mathbb{Q}\mathbf{x}_i, \mathbb{K}\mathbf{x}_k \rangle}}\end{aligned}$$

where we have assumed that $\mathbf{x}_j^\top \mathbf{A}\mathbf{x}_j$ is constant for all j. □

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^m a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^n (f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned} \dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y) \end{aligned}$$

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^m a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2} \sum_{i=1}^n (f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned}\dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y)\end{aligned}$$

By the chain rule,

$$\begin{aligned}\dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x) \dot{\theta}_t \\ &= -\underbrace{\nabla_\theta f_{\theta_t}(x) \nabla_\theta f_{\theta_t}(X)^\top}_{\mathbf{K}_t(x,X)} (f_{\theta_t}(X) - Y)\end{aligned}$$

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^m a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^n (f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned}\dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y)\end{aligned}$$

By the chain rule,

$$\begin{aligned}\dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x)\dot{\theta}_t \\ &= -\underbrace{\nabla_\theta f_{\theta_t}(x)\nabla_\theta f_{\theta_t}(X)^\top}_{\mathbf{K}_t(x,X)}(f_{\theta_t}(X) - Y)\end{aligned}$$

And $\mathbf{K}_t \xrightarrow{m\to\infty} \mathbf{K}$ (constant).

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^m a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^n (f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned} \dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y) \end{aligned}$$

By the chain rule,

$$\begin{aligned} \dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x)\dot{\theta}_t \\ &= -\underbrace{\nabla_\theta f_{\theta_t}(x)\nabla_\theta f_{\theta_t}(X)^\top}_{\mathbf{K}_t(x,X)}(f_{\theta_t}(X) - Y) \end{aligned}$$

And $\mathbf{K}_t \xrightarrow{m\to\infty} \mathbf{K}$ (constant). Then,

$$f_{\theta_t}(X) = Y + e^{-\mathbf{K}(X,X)t}(f_{\theta_0}(X) - Y)$$

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^m a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2} \sum_{i=1}^n (f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned} \dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y) \end{aligned}$$

By the chain rule,

$$\begin{aligned} \dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x) \dot{\theta}_t \\ &= -\underbrace{\nabla_\theta f_{\theta_t}(x) \nabla_\theta f_{\theta_t}(X)^\top}_{\mathbf{K}_t(x,X)} (f_{\theta_t}(X) - Y) \end{aligned}$$

And $\mathbf{K}_t \xrightarrow{m \to \infty} \mathbf{K}$ (constant). Then,

$$f_{\theta_t}(X) = Y + e^{-\mathbf{K}(X,X)t}(f_{\theta_0}(X) - Y)$$

Assuming $f_{\theta_0} \equiv 0$. Now, solve for x:

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^m a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^n (f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned}\dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y)\end{aligned}$$

By the chain rule,

$$\begin{aligned}\dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x)\dot{\theta}_t \\ &= -\underbrace{\nabla_\theta f_{\theta_t}(x)\nabla_\theta f_{\theta_t}(X)^\top}_{\mathbf{K}_t(x,X)}(f_{\theta_t}(X) - Y)\end{aligned}$$

And $\mathbf{K}_t \xrightarrow{m\to\infty} \mathbf{K}$ (constant). Then,

$$f_{\theta_t}(X) = Y + e^{-\mathbf{K}(X,X)t}(f_{\theta_0}(X) - Y)$$

Assuming $f_{\theta_0} \equiv 0$. Now, solve for x:

$$f_{\theta_t}(x) \xrightarrow{t\to\infty} \boxed{\mathbf{K}(x,X)\mathbf{K}(X,X)^{-1}Y}$$

Neural Tangent Kernel (Jacot et al. 2018)

- **Training data:** $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$
- **Model:**

$$f_\theta(x) = \sum_{j=1}^{m} a_j \sigma(w_j^\top x + b_j)$$

- **Squared loss:**

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^{n}(f_\theta(X_i) - Y_i)^2$$

- **Gradient flow:**

$$\begin{aligned}\dot{\theta}_t &:= -\nabla_\theta \mathcal{L}(\theta_t) \\ &= -\nabla_\theta f_{\theta_t}(X)^\top (f_{\theta_t}(X) - Y)\end{aligned}$$

By the chain rule,

$$\begin{aligned}\dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x)\dot{\theta}_t \\ &= -\underbrace{\nabla_\theta f_{\theta_t}(x)\nabla_\theta f_{\theta_t}(X)^\top}_{\mathbf{K}_t(x,X)}(f_{\theta_t}(X) - Y)\end{aligned}$$

And $\mathbf{K}_t \xrightarrow{m\to\infty} \mathbf{K}$ (constant). Then,

$$f_{\theta_t}(X) = Y + e^{-\mathbf{K}(X,X)t}(f_{\theta_0}(X) - Y)$$

Assuming $f_{\theta_0} \equiv 0$. Now, solve for x:

$$f_{\theta_t}(x) \xrightarrow{t\to\infty} \boxed{\mathbf{K}(x,X)\mathbf{K}(X,X)^{-1}Y}$$

This is exactly kernel regression!

Mean-Field Neural Networks (Barron Spaces)

Finite-width NN:

$$f_\theta(x) = \sum_{i=1}^{n} a_i \sigma(w_i^\top x + b_i) \tag{1}$$

where $\theta_i = (a_i, w_i, b_i) \in \mathbb{R}^{d+2}$.

Mean-Field NN:

$$f_\pi(x) = \int_{\mathbb{R}^{d+2}} a\sigma(w^\top x + b) d\pi(a, w, b)$$

where $\pi \in \mathcal{P}(\mathbb{R}^{d+2})$ is a probability measure.

Mean-Field Neural Networks (Barron Spaces)

Finite-width NN:

$$f_\theta(x) = \sum_{i=1}^{n} a_i \sigma(w_i^\top x + b_i) \qquad (1)$$

where $\theta_i = (a_i, w_i, b_i) \in \mathbb{R}^{d+2}$.

Mean-Field NN:

$$f_\pi(x) = \int_{\mathbb{R}^{d+2}} a\sigma(w^\top x + b) d\pi(a, w, b)$$

where $\pi \in \mathcal{P}(\mathbb{R}^{d+2})$ is a probability measure.

Consider the norm

$$\|f\|_{\mathcal{B}_\sigma} := \inf_{f = f_\pi} \int_{\mathbb{R}^{d+2}} |a|(1 + \|w\|_1 + |b|) d\pi$$

Theorem (Spek et al. 2023)

The space of mean-field NN with norm $\|\cdot\|_{\mathcal{B}_\sigma}$ is a reproducing kernel Banach space.

Mean-Field Neural Networks (Barron Spaces)

Finite-width NN:

$$f_\theta(x) = \sum_{i=1}^{n} a_i \sigma(w_i^\top x + b_i) \qquad (1)$$

where $\theta_i = (a_i, w_i, b_i) \in \mathbb{R}^{d+2}$.

Mean-Field NN:

$$f_\pi(x) = \int_{\mathbb{R}^{d+2}} a\sigma(w^\top x + b) d\pi(a, w, b)$$

where $\pi \in \mathcal{P}(\mathbb{R}^{d+2})$ is a probability measure.

Consider the norm

$$\|f\|_{\mathcal{B}_\sigma} := \inf_{f = f_\pi} \int_{\mathbb{R}^{d+2}} |a|(1 + \|w\|_1 + |b|) d\pi$$

Theorem (Spek et al. 2023)

The space of mean-field NN with norm $\|\cdot\|_{\mathcal{B}_\sigma}$ is a reproducing kernel Banach space.

Theorem (Bartolucci et al. 2021)

Let $\{(x_i, y_i)\}_{i=1}^n$ be training data. Then

$$\inf_{f \in \mathcal{B}_\sigma} \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \lambda \|f\|_{\mathcal{B}_\sigma}$$

admits a minimizer of the form (1).

Neural Networks vs. Kernel Methods

We want to compare kernel methods with neural networks

$$\sum_i a_i \sigma(w_i^\top x + b_i) \qquad \text{vs} \qquad \sum_i \alpha_i \kappa(x, x_i)$$

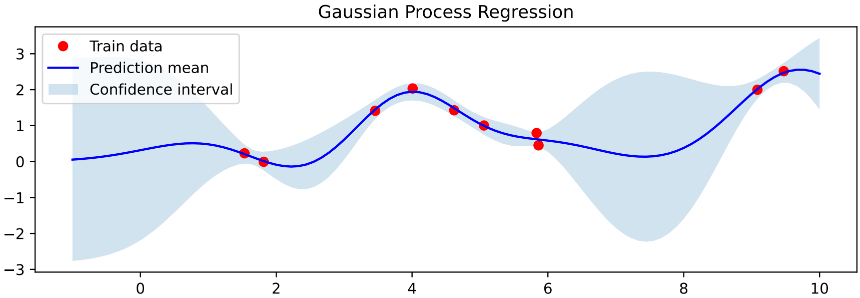
	Neural Networks	**Kernel Methods**
Universal Approximation	✓	✓
Representer Theorem	✓	✓
Compatibility	✓	✓
Interpretability	✗	**?**
Scalability	✓	**?**
Experiments	✓	**?**

Interpretability of Kernel Methods

Decomposability:

$$f(x) = \sum_i \overbrace{\alpha_i}^{\text{weight}} \underbrace{\kappa(x, x_i)}_{\text{similarity}}$$

Uncertainty Quantification:



Experiments: Kernel Methods and Digital Twins

Data: $\{y_{t,i}\}_{i=1}^{n}$.
Goal: learn the vector field $f \in \mathcal{F}$ so that $z_t \approx y_t$ with

$$\dot{z}_t = f(z_t), \qquad z_0 = y_0$$

Experiments: Kernel Methods and Digital Twins

Data: $\{y_{t,i}\}_{i=1}^{n}$.
Goal: learn the vector field $f \in \mathcal{F}$ so that $z_t \approx y_t$ with

$$\dot{z}_t = f(z_t), \qquad z_0 = y_0$$

$$\boxed{\begin{aligned} &\inf_{f \in \mathcal{F}} \quad \frac{1}{2n} \sum_{i=1}^{n} \int_0^T \|z_{t,i} - y_{t,i}\|^2 dt + \frac{\lambda}{2} \|f\|_{\mathcal{F}}^2 \\ &\quad \text{s.t. } \dot{z}_{t,i} = f(z_{t,i}), \quad z_{0,i} = y_{0,i}. \end{aligned}}$$

Experiments: Kernel Methods and Digital Twins

Data: $\{y_{t,i}\}_{i=1}^{n}$.
Goal: learn the vector field $f \in \mathcal{F}$ so that $z_t \approx y_t$ with

$$\dot{z}_t = f(z_t), \qquad z_0 = y_0$$

$$\boxed{\begin{aligned} &\inf_{f \in \mathcal{F}} \quad \frac{1}{2n} \sum_{i=1}^{n} \int_0^T \|z_{t,i} - y_{t,i}\|^2 dt + \frac{\lambda}{2} \|f\|_{\mathcal{F}}^2 \\ &\quad \text{s.t.} \;\; \dot{z}_{t,i} = f(z_{t,i}), \quad z_{0,i} = y_{0,i}. \end{aligned}}$$

NeuralODE (Chen et al. 2018)

- Model:

$$f_\theta \in \mathsf{NN}(\Theta)$$

- Regularization:

$$\|f_\theta\|_{\mathcal{F}} := \|\theta\|_2$$

KernelODE (Owhadi 2020)

- Model:

$$f \in \mathcal{H}_\kappa$$

- Regularization:

$$\|f\|_{\mathcal{F}} := \|f\|_{\mathcal{H}_\kappa}$$

Experiments: Kernel Methods and Digital Twins

Data: $\{y_{t,i}\}_{i=1}^n$.
Goal: learn the vector field $f \in \mathcal{F}$ so that $z_t \approx y_t$ with

$$\dot{z}_t = f(z_t), \qquad z_0 = y_0$$

$$\boxed{\begin{aligned} &\inf_{f\in\mathcal{F}} \quad \frac{1}{2n}\sum_{i=1}^n \int_0^T \|z_{t,i} - y_{t,i}\|^2 dt + \frac{\lambda}{2}\|f\|_{\mathcal{F}}^2 \\ &\text{s.t. } \dot{z}_{t,i} = f(z_{t,i}), \quad z_{0,i} = y_{0,i}. \end{aligned}}$$

NeuralODE (Chen et al. 2018)

$$\begin{aligned} \dot{z}_{t,i} &= f_\theta\big(z_{t,i}\big), \\ \dot{p}_{t,i} &= (z_{t,i} - y_{t,i}) - \big(Df_\theta(z_{t,i})\big)^\top p_{t,i}, \\ \theta &= -\frac{1}{\lambda n}\sum_{j=1}^n \int_0^T (\partial_\theta f_\theta(z_{s,j}))^\top p_{s,j}\, ds. \end{aligned}$$

KernelODE (Owhadi 2020),
Mean-Field NeuralODE (Jabir et al. 2021)

$$\begin{aligned} \dot{z}_{t,i} &= f\big(z_{t,i}\big), \\ \dot{p}_{t,i} &= (z_{t,i} - y_{t,i}) - \big(Df(z_{t,i})\big)^\top p_{t,i}, \\ f(\cdot) &= -\frac{1}{\lambda n}\sum_{j=1}^n \int_0^T K\big(\cdot, z_{s,j}\big)\, p_{s,j}\, ds. \end{aligned}$$

where $z_{0,i} = y_{0,i}$ and $p_{T,i} = 0$.

FitzHugh–Nagumo

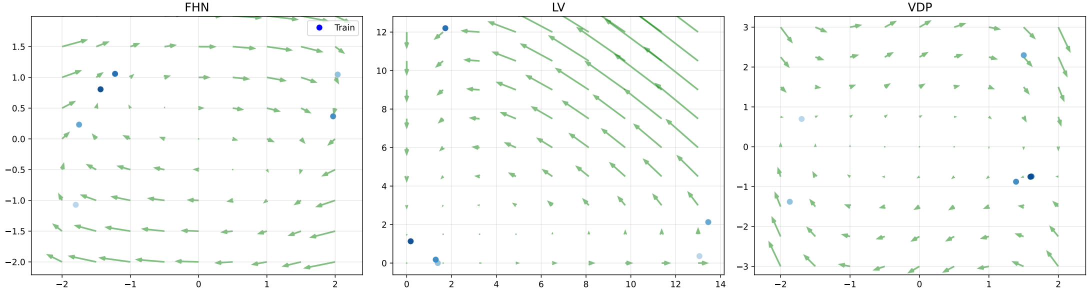
$$\dot{x}_1 = 3\left(x_1 - \frac{x_1^3}{3} + x_2\right),$$
$$\dot{x}_2 = \frac{0.2 - 3x_1 - 0.2x_2}{3}$$

Lotka–Volterra

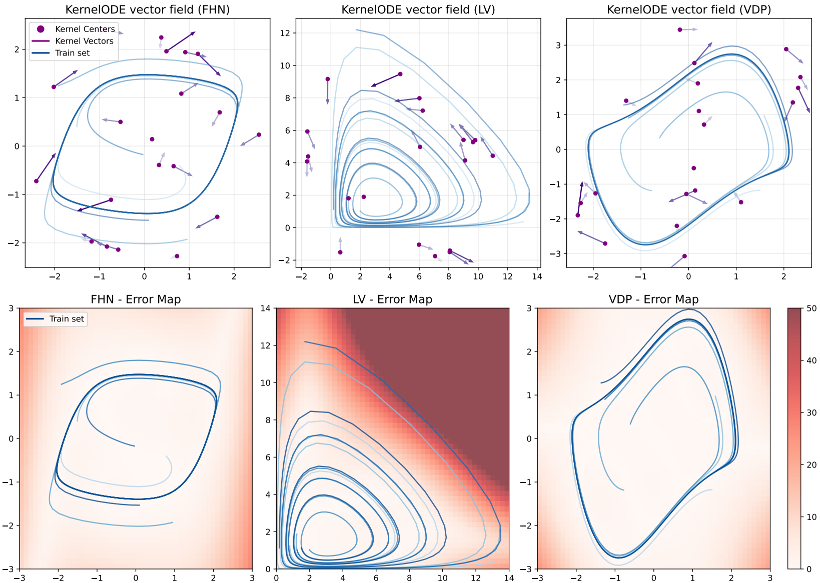
$$\dot{x}_1 = 1.5\,x_1 - x_1x_2,$$
$$\dot{x}_2 = -3\,x_2 + x_1x_2$$

Van der Pol

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = (1 - x_1^2)\,x_2 - x_1$$



Experiments: Kernel Methods and Digital Twins



Experiments: Kernel Methods and Digital Twins

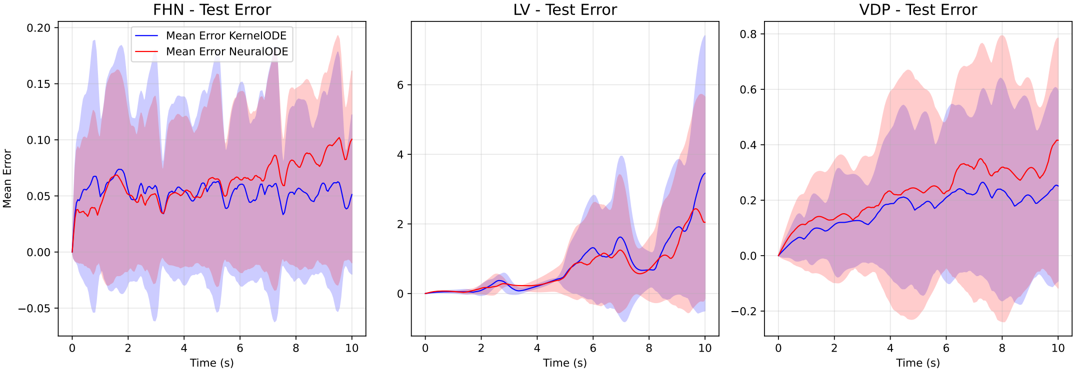


Figure: Comparison between NeuralODE and KernelODE.

Applications of Kernel Methods

Algorithms based on Kernel Methods:

- Gaussian Processes (Rasmussen et al. 2005)
- Support Vector Machines (Cortes et al. 1995)
- Kernel PCA (Schölkopf, Alexander Smola, et al. 1998)
- Kernel Mean Embeddings (Alex Smola et al. 2007)
- Neural Tangent Kernels (Jacot et al. 2018)
- Operator Learning (Batlle et al. 2023)
- Kernel-based Transformers (Choromanski et al. 2022; Peng et al. 2021)

Large Scale Applications:

- Large-scale kernel machines (Rahimi et al. 2007)
- Random Feature Attention (Peng et al. 2021): $\mathcal{O}(n^2)$ to $\mathcal{O}(nm)$.
- Rethinking Attention with Performers (Choromanski et al. 2022)
- Nyströmformer: A Nyström-Based Algorithm for Approximating Self-Attention (Xiong et al. 2021).
- Kernel Methods are Competitive for Operator Learning (Batlle et al. 2023)

References

Aronszajn, N. (1950). “Theory of Reproducing Kernels”. In: *Transactions of the American Mathematical Society* 68.3, pp. 337–404. ISSN: 0002-9947, 1088-6850. DOI: 10.1090/S0002-9947-1950-0051437-7. URL: https://www.ams.org/tran/1950-068-03/S0002-9947-1950-0051437-7/ (visited on 11/09/2025).

Bartolucci, Francesca et al. (Oct. 26, 2021). *Understanding Neural Networks with Reproducing Kernel Banach Spaces*. DOI: 10.48550/arXiv.2109.09710. arXiv: 2109.09710 [stat]. URL: http://arxiv.org/abs/2109.09710 (visited on 11/19/2025). Pre-published.

Batlle, Pau et al. (Oct. 8, 2023). *Kernel Methods Are Competitive for Operator Learning*. DOI: 10.48550/arXiv.2304.13202. arXiv: 2304.13202 [stat]. URL: http://arxiv.org/abs/2304.13202 (visited on 11/09/2025). Pre-published.

Chen, Ricky T. Q. et al. (Dec. 14, 2018). *Neural Ordinary Differential Equations*. DOI: 10.48550/arXiv.1806.07366. arXiv: 1806.07366 [cs]. URL: http://arxiv.org/abs/1806.07366 (visited on 11/09/2025). Pre-published.

Choromanski, Krzysztof et al. (Nov. 19, 2022). *Rethinking Attention with Performers*. DOI: 10.48550/arXiv.2009.14794. arXiv: 2009.14794 [cs]. URL: http://arxiv.org/abs/2009.14794 (visited on 11/08/2025). Pre-published.

Cortes, Corinna et al. (Sept. 1, 1995). “Support-Vector Networks”. In: *Machine Learning* 20.3, pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018. URL: https://doi.org/10.1007/BF00994018 (visited on 11/10/2025).

Jabir, Jean-François et al. (Mar. 16, 2021). *Mean-Field Neural ODEs via Relaxed Optimal Control*. DOI: 10.48550/arXiv.1912.05475. arXiv: 1912.05475 [math]. URL: http://arxiv.org/abs/1912.05475 (visited on 11/20/2025). Pre-published.

Jacot, Arthur et al. (Feb. 10, 2018). *Neural Tangent Kernel: Convergence and Generalization in Neural Networks*. DOI: 10.48550/arXiv.1806.07572. arXiv: 1806.07572 [cs]. URL: http://arxiv.org/abs/1806.07572 (visited on 11/09/2025). Pre-published.

Nadaraya, E. A. (Jan. 1964). “On Estimating Regression”. In: *Theory of Probability & Its Applications* 9.1, pp. 141–142. ISSN: 0040-585X, 1095-7219. DOI: 10.1137/1109020. URL: http://epubs.siam.org/doi/10.1137/1109020 (visited on 11/19/2025).

Owhadi, Houman (Oct. 27, 2020). *Do Ideas Have Shape? Idea Registration as the Continuous Limit of Artificial Neural Networks*. DOI: 10.48550/arXiv.2008.03920. arXiv: 2008.03920 [stat]. URL: http://arxiv.org/abs/2008.03920 (visited on 11/09/2025). Pre-published.

Peng, Hao et al. (Mar. 19, 2021). *Random Feature Attention*. DOI: 10.48550/arXiv.2103.02143. arXiv: 2103.02143 [cs]. URL: http://arxiv.org/abs/2103.02143 (visited on 11/09/2025). Pre-published.

Rahimi, Ali et al. (2007). “Random Features for Large-Scale Kernel Machines”. In: *Advances in Neural Information Processing Systems*. Vol. 20. Curran Associates, Inc. URL: https://papers.nips.cc/paper_files/paper/2007/hash/013a006f03dbc5392effeb8f18fda755-Abstract.html (visited on 11/08/2025).

Rasmussen, Carl Edward et al. (Nov. 23, 2005). *Gaussian Processes for Machine Learning*. The MIT Press. ISBN: 978-0-262-25683-4. DOI: 10.7551/mitpress/3206.001.0001. URL: https://direct.mit.edu/books/oa-monograph/2320/Gaussian-Processes-for-Machine-Learning (visited on 11/10/2025).

Schölkopf, Bernhard, Ralf Herbrich, et al. (2001). “A Generalized Representer Theorem”. In: *Computational Learning Theory*. Ed. by David Helmbold et al. Berlin, Heidelberg: Springer, pp. 416–426. ISBN: 978-3-540-44581-4. DOI: 10.1007/3-540-44581-1_27.

Schölkopf, Bernhard, Alexander Smola, et al. (July 1, 1998). “Nonlinear Component Analysis as a Kernel Eigenvalue Problem”. In: *Neural Computation* 10.5, pp. 1299–1319. ISSN: 0899-7667, 1530-888X. DOI: 10.1162/089976698300017467. URL: https://direct.mit.edu/neco/article/10/5/1299-1319/6193 (visited on 11/10/2025).

Smola, Alex et al. (2007). “A Hilbert Space Embedding for Distributions”. In: *Algorithmic Learning Theory*. Ed. by Marcus Hutter et al. Berlin, Heidelberg: Springer, pp. 13–31. ISBN: 978-3-540-75225-7. DOI: 10.1007/978-3-540-75225-7_5.

Spek, Len et al. (Mar. 10, 2023). *Duality for Neural Networks through Reproducing Kernel Banach Spaces*. DOI: 10.48550/arXiv.2211.05020. arXiv: 2211.05020 [math]. URL: http://arxiv.org/abs/2211.05020 (visited on 11/18/2025). Pre-published.

Watson, Geoffrey S. (1964). “Smooth Regression Analysis”. In: *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)* 26.4, pp. 359–372. ISSN: 0581572X. JSTOR: 25049340. URL: http://www.jstor.org/stable/25049340 (visited on 11/19/2025).

Xiong, Yunyang et al. (Mar. 31, 2021). *Nyströmformer: A Nyström-Based Algorithm for Approximating Self-Attention*. DOI: 10.48550/arXiv.2102.03902. arXiv: 2102.03902 [cs]. URL: http://arxiv.org/abs/2102.03902 (visited on 11/09/2025). Pre-published.

The End