A Mathematical Perspective of Machine Learning

Machine learning from the viewpoint of **numerical analysis in high dimension**

Weinan E

Princeton University

Joint work with:

Chao Ma, Lei Wu

`www.math.princeton.edu/~weinan`

Outline

Outline

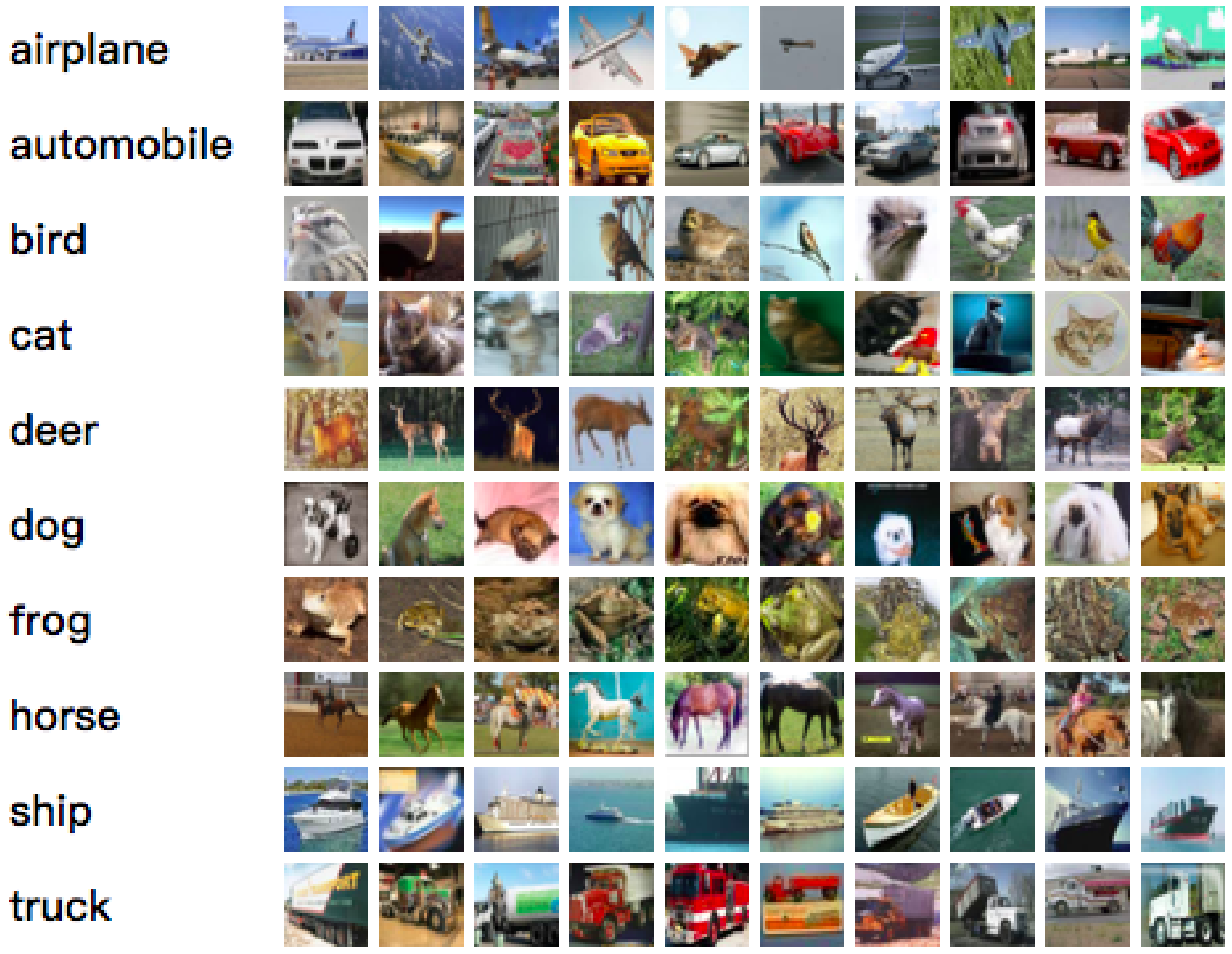
Basic example: Supervised learning

Given $S = \{(\boldsymbol{x}_j, y_j = f^*(\boldsymbol{x}_j)), j \in [n]\}$, learn f^*.

1. This is a problem about function approximation.
2. Based on finite pieces of “labeled” data
   - regression (f^* is continuous) vs classification (f^* is discrete)
   - will neglect measurement noise (not crucial for the talk)
   - assume $\boldsymbol{x}_j \in X = [0,1]^d$. d is typically quite large
   - notation: $\mu =$ the distribution of $\{\boldsymbol{x}_j\}$

In practice, one divides S into two subsets, a training set and a testing set.

Classification example: Cifar 10

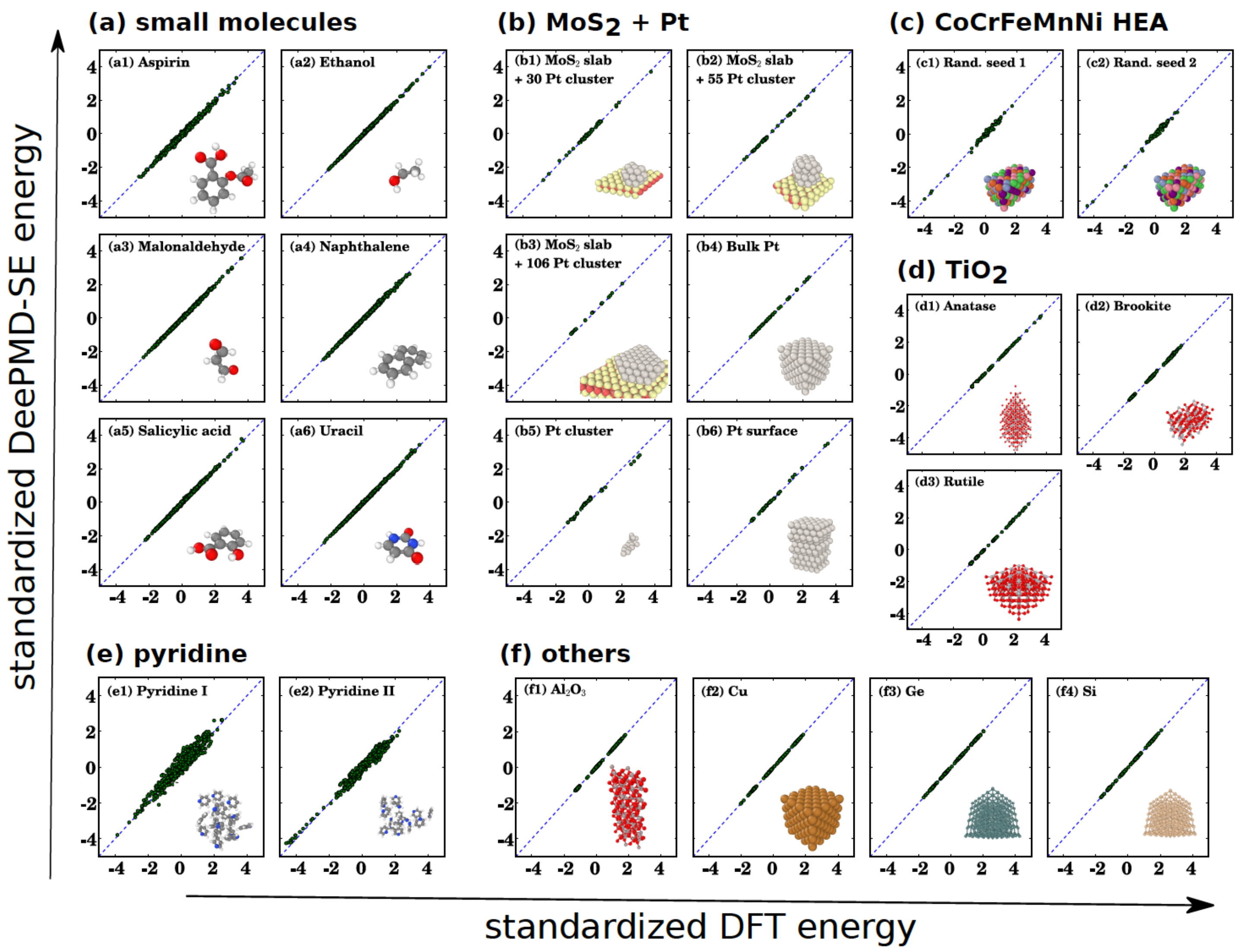


- Input: $\boldsymbol{x} \in [0,1]^d$ with $d = 32 \times 32 \times 3 = 3072$.
- Output: $f^* \in \{0, 1, \ldots, 9\}$.

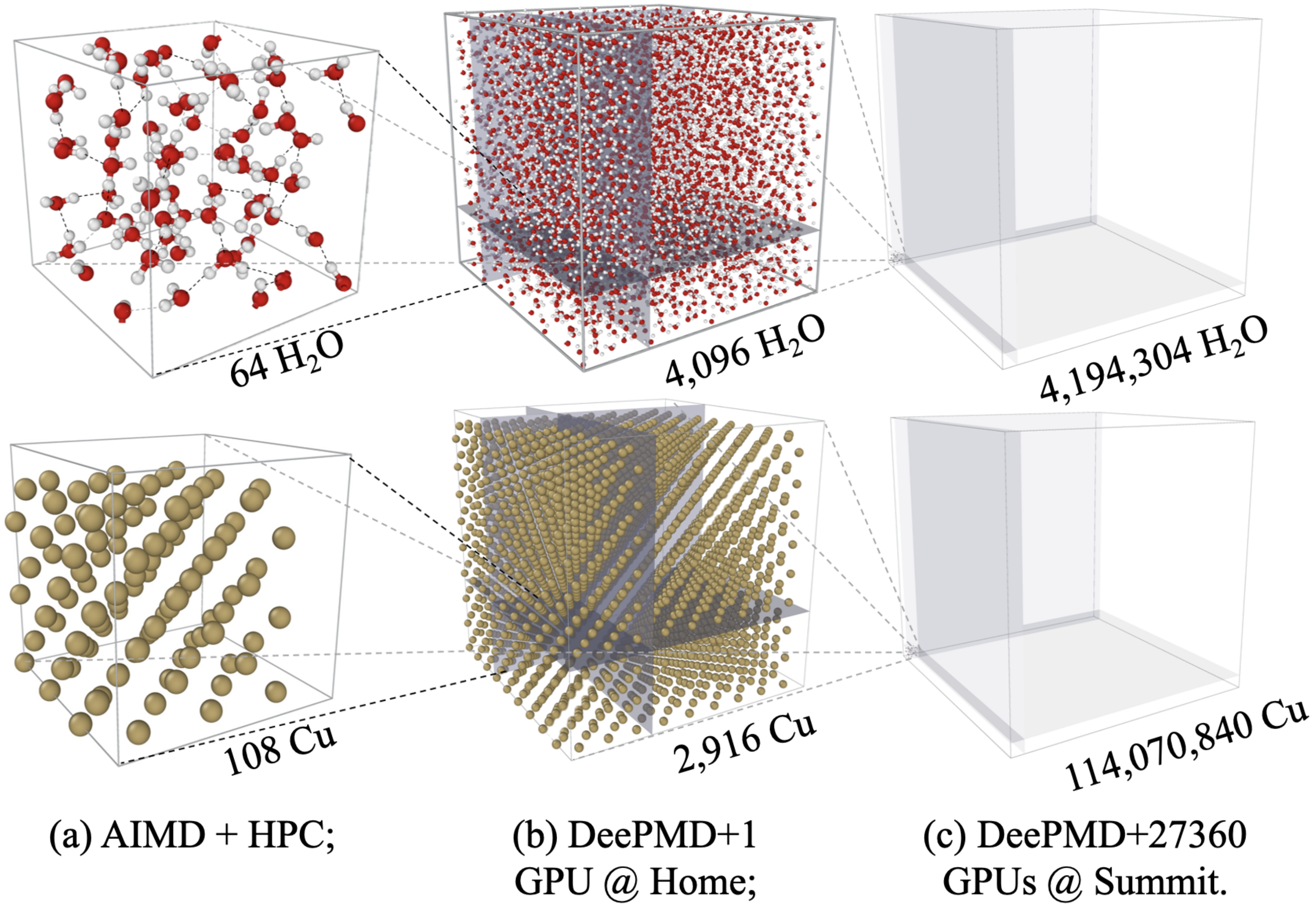
Regression example: Inter-atomic potential

http://www.deepmd.org/database/deeppot-se-data/

$$V = V(\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N)$$



86 PFLOPS DeePMD simulation of 100M atoms



D. Lu, et al, arXiv: 2004.11658; W. Jia, et al, arXiv: 2005.00223

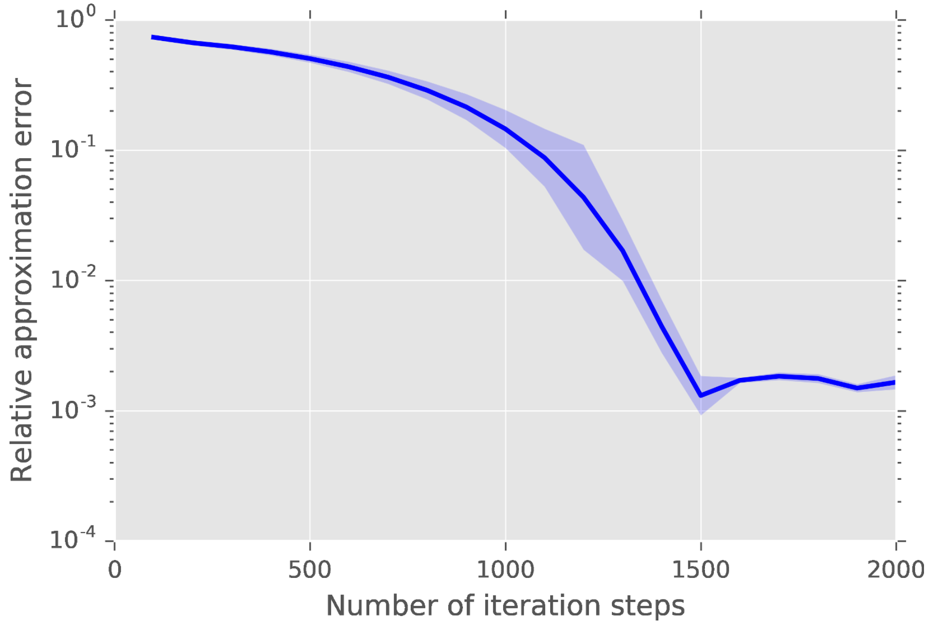
Solving PDEs in very high dimensions

HJB equation:

$$\frac{\partial u}{\partial t} - \Delta u + \lambda \|\nabla u\|_2^2 = 0, \quad u(\cdot, 0) = g(\cdot)$$

$$u(t, \boldsymbol{x}) = -\lambda^{-1} \ln\Bigg(\mathbb{E}\Big[\exp\Big(-\lambda g(\boldsymbol{x} + \sqrt{2} W_t)\Big)\Big]\Bigg).$$

Result of the DeeP BSDE method for $d = 100$:



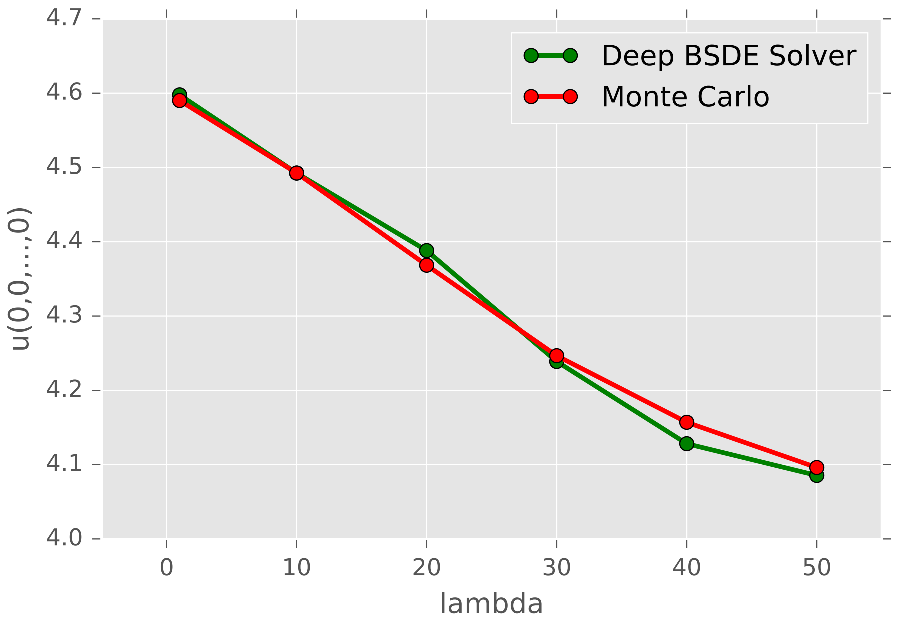


Figure: Left: Relative error of the deep BSDE method for $u(t{=}1, \boldsymbol{x}{=}(0,\ldots,0))$ when $\lambda = 1$, which achieves 0.17% in a runtime of 330 seconds. Right: Optimal cost $u(t{=}1, \boldsymbol{x}{=}(0,\ldots,0))$ against different λ.

ML in computational science and scientific computing

Attacking problems in high dimensions

1. PDEs and control problems
    - Control problems in high dimension (Han and E, 2016)
    - High dimensional nonlinear parabolic PDEs (Han, Jentzen and E, 2017)
    - Least square methods for PDEs (Sirignano and Spiliopoulos, 2018)
2. Molecular dynamics
    - Neural network models (Behler and Parrinello, 2007)
    - Kernel methods (Bartok, Payne, Kondor, Csanyi, 2010)
    - Deep Potential Molecular Dynamics (DeePMD, Zhang, Han, Wang, Car and E, 2017): *ab initio* molecular dynamics with millions of atoms
3. Quantum many-body problem
    - Schrodinger equation for spins (Carleo and Troyer, 2016)
    - Schrodinger equation for electrons (Han, Zhang and E, 2018)
    - DeepMind group (Pfau, Spencer, Matthews and Foulkes, 2019)
4. Multi-scale modeling
    - Uniformly accurate moment closure models for kinetic equations (Han, Ma, Ma and E, 2019)
    - Hydrodynamic models for polymer fluids (Lei and E 2020)

Mathematical perspective

- Error analysis (approximation theory): Good solutions do exist.
- Analysis of the training algorithm: “Work but subtle”. Why?
- Better formulation of the problem: Improve the robustness of the ML procedure

Standard procedure for supervised learning

Focus on regression problem

1. choose a hypothesis space (set of trial functions) $\mathcal{H}_m$ ($m \sim \dim(\mathcal{H}_m)$)
   - (piecewise) polynomials, wavelets, ...
   - neural network models
2. choose a loss function (to fit the data)
   - "empirical risk"
     $$\hat{\mathcal{R}}_n(f) = \frac{1}{n}\sum_j (f(\boldsymbol{x}_j) - y_j)^2 = \frac{1}{n}\sum_j (f(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j))^2$$
   - add regularization
3. choose an optimization algorithm and parameters
   - gradient descent (GD), stochastic gradient descent (SGD), ADAM, RMSprop, ...
   - hyperparameters (initialization, step size=learning rate, ...)

Objective: Minimize the "population risk" (also known as the "generalization error")

$$\mathcal{R}(f) = \mathbb{E}_{\boldsymbol{x}\sim\mu}(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 = \int_{\mathbb{R}^d} (f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 d\mu$$

Important parameters: m, n, t, d (typically interested in the case: $m, n, t \to \infty, d \gg 1$).

High dimensional integration

$$I(g) = \int_X g(\boldsymbol{x}) d\mu, \quad I_m(g) = \frac{1}{m} \sum_j g(\boldsymbol{x}_j)$$

Grid-based quadrature rules:

$$I(g) - I_m(g) \sim \frac{C(g)}{m^{\alpha/d}}$$

Appearance of $1/d$ in the exponent of m: **Curse of dimensionality (CoD)**!
If we want $m^{-\alpha/d} = 0.1$, then $m = 10^{d/\alpha} = 10^d$, if $\alpha = 1$.

Monte Carlo: $X = [0,1]^d$, $\{\boldsymbol{x}_j, j \in [m]\}$ is uniformly distributed in X.

$$\mathbb{E}(I(g) - I_m(g))^2 = \frac{\text{var}(g)}{m}, \quad \text{var}(g) = \int_X g^2(\boldsymbol{x}) d\boldsymbol{x} - \left(\int_X g(\boldsymbol{x}) d\boldsymbol{x}\right)^2$$

The $O(1/\sqrt{m})$ rate is (almost) the best we can hope for.

However, var(g) can be very large in high dimension. Variance reduction!

Approximation and estimation errors

Want to understand $f^* - \hat{f}$, where $\hat{f}$ = the output of the ML model.

$f_m = \text{argmin}_{f \in \mathcal{H}_m} \mathcal{R}(f) =$ “best approx” to f^* in $\mathcal{H}_m$.

Decomposition of the error:

$$f^* - \hat{f} = f^* - f_m + f_m - \hat{f}$$

- $f^* - f_m$ is the *approximation error*, due entirely to the choice of the hypothesis space
- $f_m - \hat{f}$ is the *estimation error* — additional error caused by the fact that we only have a finite dataset

Approximation error: Basics

How do we approximate functions?

- polynomials, piecewise polynomials, splines
- Fourier,wavelets
- other special basis functions (e.g. atomic orbitals)

$$\|f - f_m\|_{L^2(X)} \leq C_0 m^{-\alpha/d} \|f\|_{H^\alpha(X)}$$

CoD!

Estimation error

Since we can only work with a finite dataset, what happens to the solution outside of the dataset?

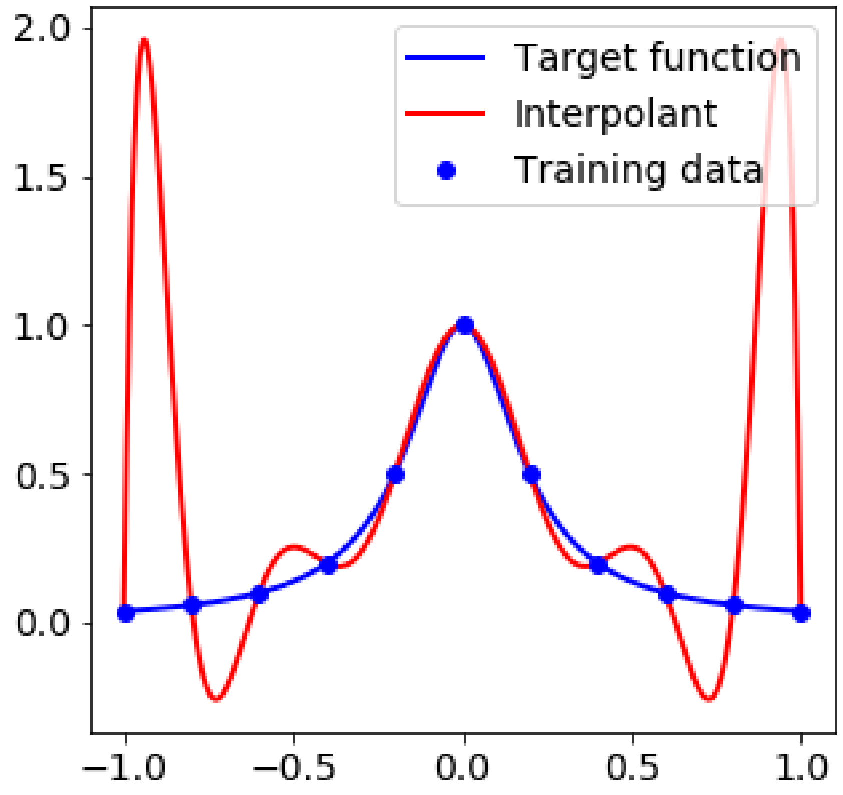


Figure: The Runge phenomenon: $f^*(x) = \frac{1}{1+25x^2}$

Generalization gap = difference between training and testing errors

$$\text{"Generalization gap"} = |\mathcal{R}(\hat{f}) - \hat{\mathcal{R}}_n(\hat{f})| = |I(g) - I_n(g)|$$

$$\mathcal{R}(f) = \int_{\mathbb{R}^d} (f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 d\mu, \quad \hat{\mathcal{R}}_n(f) = \frac{1}{n}\sum_j (f(\boldsymbol{x}_j) - f^*(\boldsymbol{x}_j))^2$$

$$I(g) = \int g(\boldsymbol{x}) d\mu, \quad I_n(g) = \frac{1}{n}\sum_j g(\boldsymbol{x}_j), \quad g(\boldsymbol{x}) = (\hat{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2$$

Difficulty: $\hat{f}$ is highly correlated with $\{\boldsymbol{x}_j\}$.

Should NOT expect: generalization gap $= O(1/\sqrt{n})$ to hold automatically.

If hypothesis space = space of all Lipschitz-1 functions:

$$\text{generalization gap} \sim \frac{1}{n^{1/d}}$$

This gives rise to **CoD for the size of the dataset**.

Rademacher complexity

Let $\mathcal{H}$ be a set of functions, and $S = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ be a set of data points. The Rademacher complexity of $\mathcal{H}$ with respect to S is defined as

$$\mathrm{Rad}_S(\mathcal{H}) = \frac{1}{n}\mathbb{E}_\xi \left[\sup_{h\in\mathcal{H}} \sum_{i=1}^n \xi_i h(\boldsymbol{x}_i) \right],$$

where $\{\xi_i\}_{i=1}^n$ are i.i.d. random variables taking values ± 1 with equal probability.

Theorem (Rademacher complexity and the generalization gap)

For any $\delta \in (0,1)$, with probability at least $1-\delta$ over the random samples $S = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$,

$$\sup_{h\in\mathcal{H}} \left| \mathbb{E}_{\boldsymbol{x}}\left[h(\boldsymbol{x})\right] - \frac{1}{n}\sum_{i=1}^n h(\boldsymbol{x}_i) \right| \le 2\,\mathrm{Rad}_S(\mathcal{H}) + \sup_{h\in\mathcal{H}} \|h\|_\infty \sqrt{\frac{\log(2/\delta)}{2n}}.$$

$$\sup_{h\in\mathcal{H}} \left| \mathbb{E}_{\boldsymbol{x}}\left[h(\boldsymbol{x})\right] - \frac{1}{n}\sum_{i=1}^n h(\boldsymbol{x}_i) \right| \ge \frac{1}{2}\mathrm{Rad}_S(\mathcal{H}) - \sup_{h\in\mathcal{H}} \|h\|_\infty \sqrt{\frac{\log(2/\delta)}{2n}}.$$

Two types of machine learning models

(1). Models that suffer from CoD:

$$\text{generalization error} = O(m^{-\alpha/d}) \text{ and/or } O(n^{-\beta/d})$$

- piecewise polynomial approximation
- wavelets with fixed wavelet basis

(2). Models that don't suffer from CoD: For example

$$\text{generalization error} = \int (\hat{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2 \mu(\boldsymbol{x}) \sim \gamma_1(f^*)/m + \gamma_2(f^*)/\sqrt{n}$$

These are "Monte-Carlo-like" bounds, γ_1 and γ_2 play the role of variance in Monte Carlo.

- random feature models
- two-layer neural network models
- deep residual neural network models

There is also the possibility of CoD from the optimization algorithm.

Outline

General procedure

Given a machine learning model:

- Approximation error:

$$\inf_{f\in\mathcal{H}_m} \mathcal{R}(f) = \inf_{f\in\mathcal{H}_m} \|f-f^*\|^2_{L^2(d\mu)} \lesssim \frac{\Gamma(f^*)^2}{m}$$

  Identify $\Gamma(f^*)$.

  - Direct and inverse approximation theorems

- Estimation error (generalization gap): $\mathcal{H}_Q = \{f, \Gamma(f) \le Q\}$. Then we have

$$\mathrm{Rad}_S(\mathcal{H}_Q) \lesssim \frac{Q}{\sqrt{n}}$$

Combined:

$$\mathcal{R}(\hat{f}) \lesssim \frac{\Gamma(f^*)^2}{m} + \frac{\Gamma(f^*)}{\sqrt{n}}$$

1. Random feature model

$\{\phi(\cdot;\boldsymbol{w})\}$: collection of random features, e.g. $\phi(\boldsymbol{x},\boldsymbol{w}) = \sigma(\boldsymbol{w}^T\boldsymbol{x})$.
π: prob distribution of the random variable $\boldsymbol{w}$.

Hypothesis space: Given any realization $\{\boldsymbol{w}_j\}_{j=1}^m$, i.i.d. with distribution π

$$\mathcal{H}_m(\{\boldsymbol{w}_j\}) = \{f_m(\boldsymbol{x},\boldsymbol{a}) = \frac{1}{m}\sum_{j=1}^m a_j\phi(\boldsymbol{x};\boldsymbol{w}_j)\}.$$

Consider functions of the form

$$\mathcal{F}_p = \{f : f(\boldsymbol{x}) = \int a(\boldsymbol{w})\phi(\cdot;\boldsymbol{w})\pi(\boldsymbol{w})\}, \|f\|_{\mathcal{F}_p} := (\mathbb{E}[|a(\boldsymbol{w})|^p])^{1/p} < \infty\}$$

$\mathcal{F}_2$ is the same as the reproducing kernel Hilbert space (RKHS) with kernel:

$$k(\boldsymbol{x},\boldsymbol{x}') = \mathbb{E}_{\boldsymbol{w}\sim\pi}[\phi(\boldsymbol{x};\boldsymbol{w})\phi(\boldsymbol{x}';\boldsymbol{w})]$$

Direct and Inverse Approximation Theorem

Theorem (Direct Approximation Theorem)

Assume $\|f\|_{\mathcal{F}_\infty} < \infty$. *Then for any* $\delta \in (0,1)$, *with probability at least* $1-\delta$ *over the random sampling of* $\{\boldsymbol{w}_j\}_{j=1}^m$, *there exists* $\boldsymbol{a} \in \mathbb{R}^m$ *such that*

$$\mathcal{R}(\boldsymbol{a}) \leq \frac{\|f\|_{\mathcal{F}_\infty}^2}{m}\left(1+\log(2/\delta)\right).$$

Moreover, $\sup_{j\in[m]} |a_j| \leq \|f\|_{\mathcal{F}_\infty}$.

Theorem (Inverse Approximation Theorem)

Assume that ϕ *is continuous and bounded, and supp*(π) *is compact. Let* $(\boldsymbol{w}_j)_{j=1}^\infty$ *be a sequence of i.i.d. samples of* π. *Assume that there exist a sequence* $(a_j)_{j=0}^\infty$ *with* $\sup_j |a_j| \leq C$, *such that*

$$\lim_{m\to\infty} \frac{1}{m}\sum_{j=1}^m a_j \sigma(\boldsymbol{w}_j \cdot \boldsymbol{x}) = f^*(\boldsymbol{x}),$$

for all $\boldsymbol{x} \in [0,1]^d$. *Then* $\|f^*\|_{\mathcal{F}_\infty} \leq C$ *and there exists* $a^*(\cdot) : \Omega \mapsto \mathbb{R}$ *such that*

$$f^*(\boldsymbol{x}) = \int_\Omega a^*(\boldsymbol{w})\sigma(\boldsymbol{w}\cdot\boldsymbol{x})d\pi(\boldsymbol{w}), \quad a.s.$$

Complexity estimates

Theorem

Let $\mathcal{F}_2(Q) = \{f \in \mathcal{F}_2, \|f\|_{\mathcal{F}_2} \leq Q\}$. Then we have

$$\mathrm{Rad}_S(\mathcal{F}_2(Q)) \leq Q\sqrt{\frac{\ln(2d)}{n}}$$

A priori estimates of the regularized model

$$\mathcal{L}_{n,\lambda}(\boldsymbol{a}) = \hat{\mathcal{R}}_n(\boldsymbol{a}) + \frac{\lambda}{\sqrt{n}}\frac{\|\boldsymbol{a}\|}{\sqrt{m}},$$

Consider the regularized estimator

$$\hat{\boldsymbol{a}}_{n,\lambda} = \text{argmin } \mathcal{L}_{n,\lambda}(\boldsymbol{a})$$

Theorem

Assume that $\|f^*\|_\infty \leq 1$. There exist a constant C_0, such that for any $\delta > 0$, if $\lambda \geq C_0, m \geq \log^2(n/\delta)$, then with probability at least $1-\delta$ over the choice of training set, we have

$$\mathcal{R}(\hat{\boldsymbol{a}}_n) \lesssim \frac{1}{m}\|f^*\|^2_{\mathcal{F}_2} + \frac{\|f^*\|_{\mathcal{F}_2}}{\sqrt{n}} + \sqrt{\frac{\log(n\|f^*\|_{\mathcal{F}_\infty}/\delta)}{n}} + \frac{\log^2(n/\delta)}{m^2}\|f^*\|^2_{\mathcal{F}_\infty}.$$

2. Two-layer neural network model: Barron spaces

$$\mathcal{H}_m = \{f_m(\boldsymbol{x}) = \frac{1}{m}\sum_j a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x})\}$$

Consider the function $f : X = [0,1]^d \mapsto \mathbb{R}$ of the following form

$$f(\boldsymbol{x}) = \int_\Omega a\sigma(\boldsymbol{w}^T\boldsymbol{x})\rho(da, d\boldsymbol{w}) = \mathbb{E}_\rho[a\sigma(\boldsymbol{w}^T\boldsymbol{x})]\}, \quad \boldsymbol{x} \in X$$

$\Omega = \mathbb{R}^1 \times \mathbb{R}^{d+1}$, ρ is a probability distribution on Ω.

$$\|f\|_{\mathcal{B}} = \inf_{\rho \in P_f} \left(\mathbb{E}_\rho[a^2\|\boldsymbol{w}\|_1^2]\right)^{1/2}$$

where $P_f := \{\rho : f(\boldsymbol{x}) = \mathbb{E}_\rho[a\sigma(\boldsymbol{w}^T\boldsymbol{x})]\}$.

$$\mathcal{B} = \{f \in C^0 : \|f\|_{\mathcal{B}} < \infty\}$$

Barron space and RKHS

Equivalent formulation (taking conditional expectation with respect to $\boldsymbol{w}$):

$$f^*(\boldsymbol{x}) = \int a(\boldsymbol{w})\sigma(\boldsymbol{w}^T\boldsymbol{x})\pi(d\boldsymbol{w}), \quad \boldsymbol{x} = (\boldsymbol{x}, 1)$$

Define:

$$k_\pi(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{\boldsymbol{w}\sim\pi}\sigma(\boldsymbol{w}^T\boldsymbol{x})\sigma(\boldsymbol{w}^T\boldsymbol{x}')$$

We can write

$$\mathcal{B} = \bigcup_\pi \mathcal{H}_{k_\pi}$$

Shallow neural network can be understood as kernel method with adaptive (learned) kernel.

The ability to learn the right kernel is VERY important.
For example, SVM would be perfect if the right kernel was known.

Theorem (Direct Approximation Theorem)

There exists an absolute constant C_0 such that

$$\|f - f_m\|_{L^2(X)} \leq \frac{C_0 \|f\|_{\mathcal{B}}}{\sqrt{m}}$$

Theorem (Inverse Approximation Theorem)

Let

$$\mathcal{N}_C \overset{def}{=} \{ \frac{1}{m} \sum_{k=1}^{m} a_k \sigma(\boldsymbol{w}_k^T \boldsymbol{x}) : \frac{1}{m} \sum_{k=1}^{m} |a_k|^2 \|\boldsymbol{w}_k\|_1^2 \leq C^2, m \in \mathbb{N}^+ \}.$$

Let f^* be a continuous function. Assume there exists a constant C and a sequence of functions $f_m \in \mathcal{N}_C$ such that

$$f_m(\boldsymbol{x}) \to f^*(\boldsymbol{x})$$

for all $\boldsymbol{x} \in X$, then there exists a probability distribution ρ^* on Ω, such that

$$f^*(\boldsymbol{x}) = \int a\sigma(\boldsymbol{w}^T \boldsymbol{x}) \rho^*(da, d\boldsymbol{w}),$$

for all $\boldsymbol{x} \in X$ and $\|f^*\|_{\mathcal{B}} \leq C$.

Complexity estimates

Theorem (Bach, 2017)

Let $\mathcal{F}_Q = \{f \in \mathcal{B}, \|f\|_{\mathcal{B}} \le Q\}$. *Then we have*

$$\mathrm{Rad}_S(\mathcal{F}_Q) \le 2Q\sqrt{\frac{2\ln(2d)}{n}}$$

A priori estimates for regularized model

$$\mathcal{L}_n(\theta) = \hat{\mathcal{R}}_n(\theta) + \lambda\sqrt{\frac{\log(2d)}{n}}\|\theta\|_{\mathcal{P}}, \qquad \hat{\theta}_n = \text{argmin}\ \mathcal{L}_n(\theta)$$

where the path norm is defined by:

$$\|\theta\|_{\mathcal{P}} = \left(\frac{1}{m}\sum_{k=1}^{m}|a_k|^2\|\boldsymbol{w}_k\|_1^2\right)^{1/2}$$

Theorem (E, Ma, Wu, 2018)

Assume that the target function $f^*: X \mapsto [0,1] \in \mathcal{B}$. There exist constants C_0, C_1, C_2, such that for any $\delta > 0$, if $\lambda \geq C_0$, then with probability at least $1-\delta$ over the choice of training set, we have

$$\mathcal{R}(\hat{\theta}_n) \leq C_1\left(\frac{\|f^*\|_{\mathcal{B}}^2}{m} + \|f^*\|_{\mathcal{B}}\sqrt{\frac{\log(2d)}{n}}\right) + C_2\sqrt{\frac{\log(4C_2/\delta)+\log(n)}{n}}.$$

Typical results in ML literature: a posteriori estimates

$$\mathcal{R}(\hat{\theta}_n) - \hat{\mathcal{R}}_n(\hat{\theta}_n) \leq C_1\frac{\|\hat{\theta}\|}{\sqrt{n}}$$

Outline

Issues

- The optimization problem: Can we make the training error small? How fast?
  The loss function is non-convex.
  Possible CoD in time?
- The generalization problem: Can we make the testing error small?
  Global optimum is non-unique (Cooper: Global minimum forms a manifold of dimension $m - n$)
  Which one is picked? Does it generalize well?
  Implicit regularization?

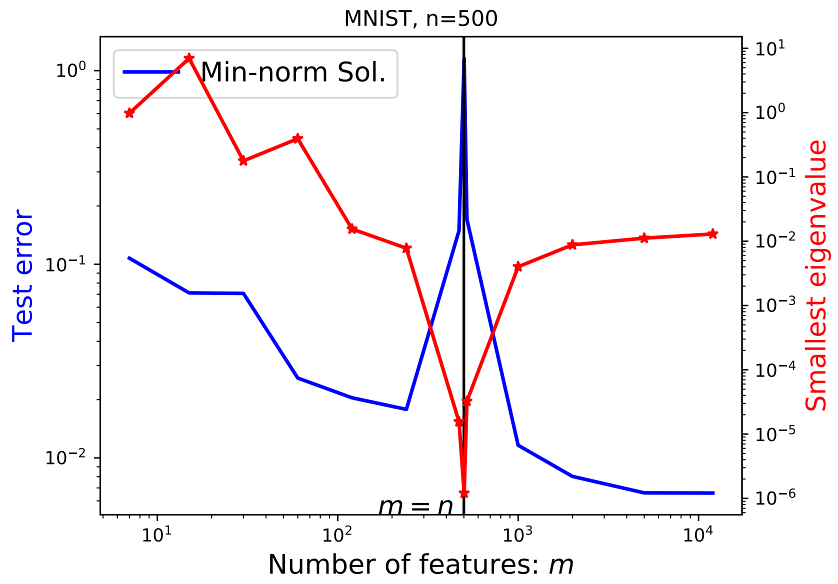
Gradient descent for the random feature model

$$f_m(\boldsymbol{x}, t) = \frac{1}{m} \sum_j a_j(t) \sigma(\boldsymbol{w}_j^T \boldsymbol{x}), \; \theta(t) = \{a_j(t), j \in [m]\}$$

$$\frac{d}{dt} a_j(t) = -\nabla_{a_j} \hat{\mathcal{R}}_n(\theta) = -\frac{1}{m} \sum_k \hat{K}(\boldsymbol{w}_j, \boldsymbol{w}_k) a_k(t) + \tilde{f}(\boldsymbol{w}_j)$$

$$\hat{K}(\boldsymbol{w}, \tilde{\boldsymbol{w}}) = \frac{1}{n} \sum_j \sigma(\boldsymbol{w}^T \boldsymbol{x}_j) \sigma(\tilde{\boldsymbol{w}}^T \boldsymbol{x}_j), \quad \tilde{f}(\boldsymbol{w}) = \frac{1}{n} \sum_j f^*(\boldsymbol{x}_j) \sigma(\boldsymbol{w}^T \boldsymbol{x}_j)$$

Resonance and slow deterioration



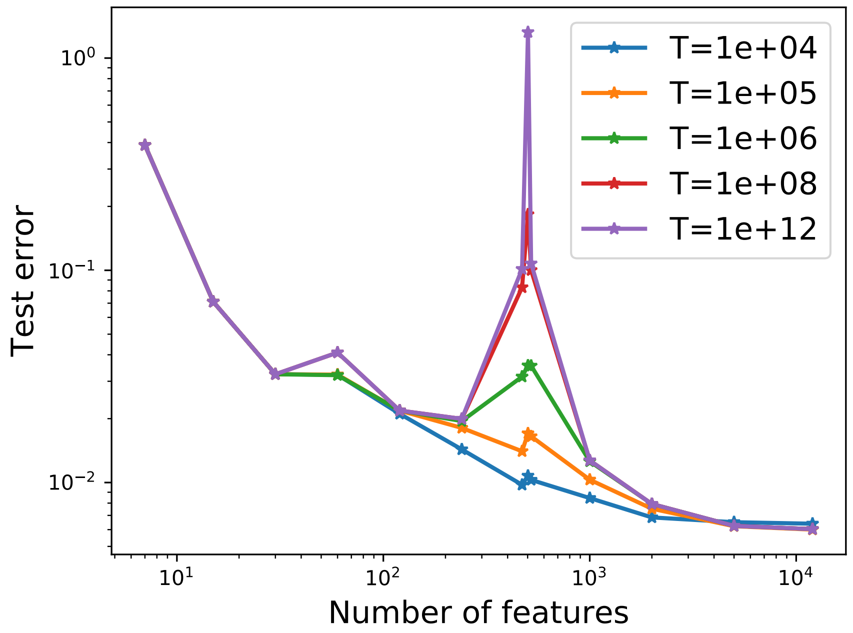


Figure: **Left:** Test error as a function of m for $n = 500$. **Right:** The test error of GD solutions obtained by running different number of iterations.

The "double descent" phenomenon (Saxe, Belkin et al)

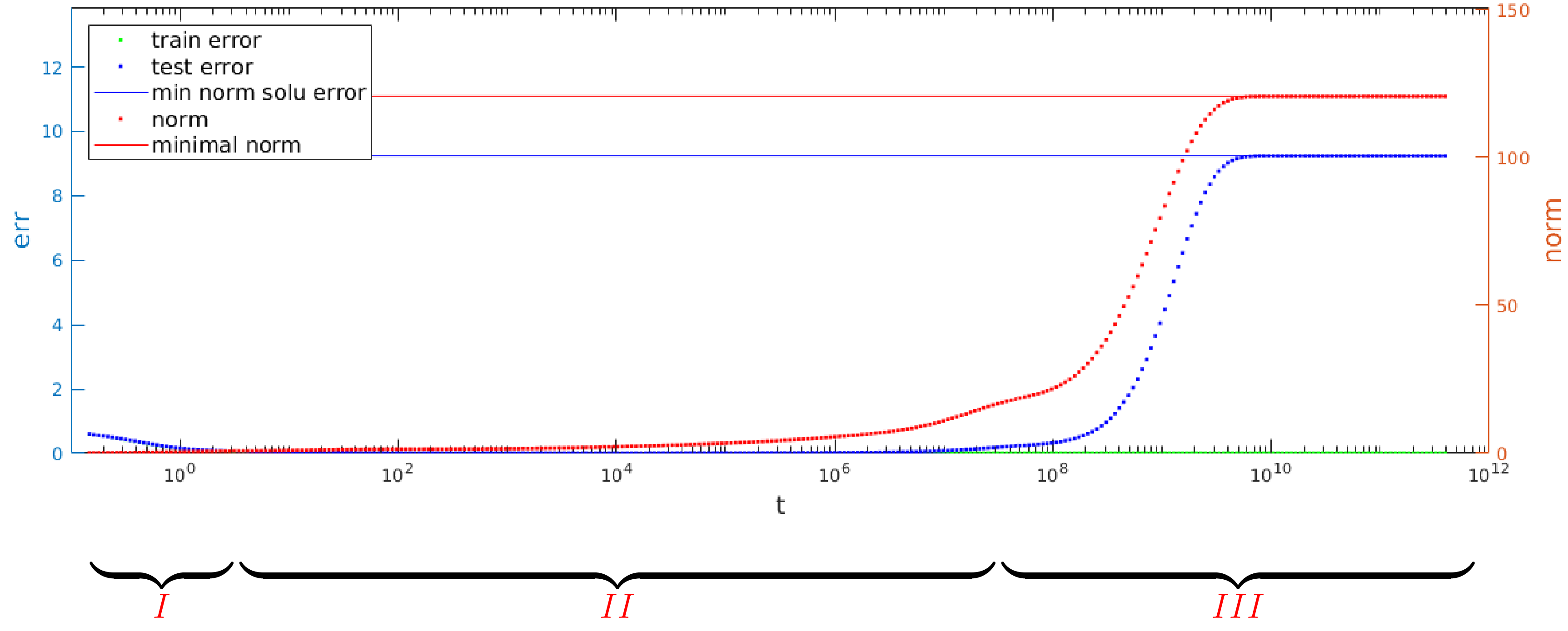
The large test error is caused by very small eigenvalues of the Gram matrix, which also lead to slow convergence.

Theorem

Let $\Phi = \sigma(X^T B)$, $\{\lambda_i\}$ *be the singular values of* Φ, *and* $\hat{\lambda}_i = \frac{\lambda_i}{n}$. *Let* $f^*(\boldsymbol{x}) = \psi_1(\boldsymbol{x})$, *the first eigenfunction of the kernel operator. There exists constant* C *s.t., for any* $\delta > 0$, *with prob no less than* $1-\delta$,

$$\|\hat{f}_t - f^*\| \leq e^{-\hat{\lambda}_1^2 t} + C\left(1 + \hat{\lambda}_{\lfloor\sqrt{n}\rfloor}^{-2}\log\frac{2}{\delta}\right)^{-1/2}\left(\frac{1}{\sqrt{n}} + M n^{-\frac{1}{4}} d(t)\right),$$

where $M = \int \|\sigma(B^T\boldsymbol{x})\|^2 \pi(d\boldsymbol{x})$, *and* $d(t) = \min\left\{\sqrt{t}, \hat{\lambda}_{\lfloor\sqrt{n}\rfloor+1} t, \hat{\lambda}_n^{-1}\right\}$.



Degeneracy of neural network models to random feature models in the over-parametrized regime

$$f_m(\boldsymbol{x};\boldsymbol{a},\boldsymbol{B}) = \sum_{j=1}^{m} a_j \sigma(\boldsymbol{b}_j^T\boldsymbol{x}) = \boldsymbol{a}^T\sigma(\boldsymbol{B}\boldsymbol{x}), \tag{1}$$

Theorem

Let $\lambda_n = \lambda_{\min}(K)$ and assume $\beta = 0$. Denote by $f_m(\boldsymbol{x};\tilde{\boldsymbol{a}}(t),\boldsymbol{B}_0))$ the solutions of GD dynamics for the random feature model. For any $\delta \in (0,1)$, assume that $m \gtrsim n^2\lambda_n^{-4}\delta^{-1}\ln(n^2\delta^{-1})$. Then with probability at least $1-6\delta$ we have

$$\hat{\mathcal{R}}_n(\boldsymbol{a}(t),\boldsymbol{B}(t)) \le e^{-m\lambda_n t}\hat{\mathcal{R}}_n(\boldsymbol{a}(0),\boldsymbol{B}(0)) \tag{2}$$

$$\sup_{\boldsymbol{x}\in\mathcal{S}^{d-1}} |f(\boldsymbol{x};\boldsymbol{a}(t),\boldsymbol{B}(t)) - f(\boldsymbol{x};\tilde{\boldsymbol{a}}(t),\boldsymbol{B}_0)| \lesssim \frac{(1+\sqrt{\ln(\delta^{-1})})^2\lambda_n^{-1}}{\sqrt{m}}. \tag{3}$$

Observation: Time scale separation

$$\dot{a}_j(t) \sim O(\|\boldsymbol{b}_j\|) = O(1) \tag{4}$$

$$\dot{\boldsymbol{b}}_j(t) \sim O(|a_j|) = O\left(\frac{1}{\lambda_n m}\right) \tag{5}$$

The dynamics of $\boldsymbol{b}$ is effectively frozen.

Outline

Looking for “well-posed” formulations of machine learning

1. The continuous formulation of the machine learning model
   - representation of functions: integral transform representation and flow-based representation
   - the variational problem for minimizing the population risk (loss function)
   - gradient flow for the variational problem (training, a PDE-like problem)
2. Discretization
   - spectral methods
   - particle methods, smoothed particle methods (analog of Monte Carlo)

Remarks:

- Conventional neural network models/algorithms can be recovered this way
- Continuous formulation-based models are easier to analyze and tend to be more robust in practice

Similar philosophy in image processing

Example: Denoising

- constructing various special purpose filters, e.g. wavelet-based, curvelet-based
- start with a continuous mathematical problems (Mumford-Shah, Rudin-Osher-Fatemi), and then discretize and optimize

$$I(u) = \int_\Omega ((u-f)^2 + \lambda|\nabla u|) d\boldsymbol{x}$$

f =original image, u= denoised image.

Representing functions: An illustrative example

Traditional approach:

$$f(\boldsymbol{x}) = \int_{\mathbb{R}^d} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} d\boldsymbol{\omega}, \quad f_m(\boldsymbol{x}) = \frac{1}{m}\sum_j a(\boldsymbol{\omega}_j) e^{i(\boldsymbol{\omega}_j,\boldsymbol{x})}$$

$\{\boldsymbol{\omega}_j\}$ is a fixed grid, e.g. uniform.

$$\|f - f_m\|_{L^2(X)} \le C_0 m^{-\alpha/d} \|f\|_{H^\alpha(X)}$$

"New" approach:

$$f(\boldsymbol{x}) = \int_{\mathbb{R}^d} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} \pi(d\boldsymbol{\omega}) = \mathbb{E}_{\boldsymbol{\omega}\sim\pi} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})}$$

where π is a probability measure on $\mathbb{R}^d$. Let $\{\boldsymbol{\omega}_j\}$ be an i.i.d. sample of π.

$$\mathbb{E}|f(\boldsymbol{x}) - \frac{1}{m}\sum_{j=1}^m a(\boldsymbol{\omega}_j) e^{i(\boldsymbol{\omega}_j,\boldsymbol{x})}|^2 = \frac{\mathrm{var}(f)}{m}$$

where

$$\mathrm{var}(f) = \mathbb{E}_{\boldsymbol{\omega}\sim\pi}|a(\boldsymbol{\omega})|^2 - f(\boldsymbol{x})^2$$

$f(\boldsymbol{x}) = \int_{\mathbb{R}^d} a(\boldsymbol{\omega}) e^{i(\boldsymbol{\omega},\boldsymbol{x})} \pi(d\boldsymbol{\omega})$ can be regarded as a continuous version of two-layer neural network function with activation function σ defined by $\sigma(z) = e^{iz}$.

Integral transform-based representation and the variational problem

Now consider functions represented in the form:

$$f(\boldsymbol{x},\theta) = \int_{\mathbb{R}^d} a(\boldsymbol{w})\sigma(\boldsymbol{w}^T\boldsymbol{x})\pi(d\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{w}\sim\pi} a(\boldsymbol{w})\sigma(\boldsymbol{w}^T\boldsymbol{x})$$

where θ denotes the parameter to be optimized in order to get the best approximation of some target function f^*. θ can either be $a(\cdot)$ which corresponds to a feature-based model, or the pair $(a(\cdot),\pi(\cdot))$ in which case we get a two-layer neural network-like model.
One can also write the above equivalently as

$$f(\boldsymbol{x},\theta) = \mathbb{E}_{(a,\boldsymbol{w})\sim\rho} a\sigma(\boldsymbol{w}^T\boldsymbol{x})$$

where $\rho(da, d\boldsymbol{w}) = \pi(d\boldsymbol{w})\delta(a - a(\boldsymbol{w}))da$. In this case, $\theta = \rho$.

Given a target function f^*, the variational problem for minimizing the population risk becomes $\min \mathcal{R}$ where

$$\mathcal{R}(\theta) = \mathbb{E}_{\boldsymbol{x}\sim\mu}(f(\boldsymbol{x},\theta) - f^*(\boldsymbol{x}))^2$$

The continuous formulation: Gradient flows

Recall the population risk: $\mathcal{R}(f) = \mathbb{E}_{\boldsymbol{x}\sim\mu}(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2$ = "free energy"

$$f(\boldsymbol{x}) = \int a(\boldsymbol{w})\sigma(\boldsymbol{w}^T\boldsymbol{x})\pi(d\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{w}\sim\pi} a(\boldsymbol{w})\sigma(\boldsymbol{w}^T\boldsymbol{x})$$

Follow Halperin and Hohenberg (1977)

- $a =$ non-conserved, use "model A" dynamics (Allen-Cahn):

$$\frac{\partial a}{\partial t} = -\frac{\delta \mathcal{R}}{\delta a}$$

- $\pi =$ conserved (probability density), use "model B" (Cahn-Hilliard):

$$\frac{\partial \pi}{\partial t} + \nabla \cdot \mathbf{J} = 0$$

$$\mathbf{J} = \pi \boldsymbol{v},\ \boldsymbol{v} = -\nabla V,\ V = \frac{\delta \mathcal{R}}{\delta \pi}.$$

Gradient flow for the feature-based model

Fix π, optimize over a.

$$\partial_t a(\boldsymbol{w}, t) = -\frac{\delta \mathcal{R}}{\delta a}(\boldsymbol{w}, t) = -\int a(\tilde{\boldsymbol{w}}, t) K(\boldsymbol{w}, \tilde{\boldsymbol{w}}) \pi(d\tilde{\boldsymbol{w}}) + \tilde{f}(\boldsymbol{w})$$

$$K(\boldsymbol{w}, \tilde{\boldsymbol{w}}) = \mathbb{E}_{\boldsymbol{x}}[\sigma(\boldsymbol{w}^T \boldsymbol{x}) \sigma(\tilde{\boldsymbol{w}}^T \boldsymbol{x})], \quad \tilde{f}(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{x}}[f^*(\boldsymbol{x}) \sigma(\boldsymbol{w}^T \boldsymbol{x})]$$

This is an integral equation with a symmetric positive definite kernel.

Decay estimates due to convexity: Let $f^*(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{w} \sim \pi} a^*(\boldsymbol{w}) \sigma(\boldsymbol{w}^T \boldsymbol{x})$,

$$I(t) = \frac{1}{2} \|a(\cdot, t) - a^*(\cdot)\|^2 + t(\mathcal{R}(a(t)) - \mathcal{R}(a^*))$$

Then we have

$$\frac{dI}{dt} \leq 0, \quad \mathcal{R}(a(t)) \leq \frac{C_0}{t}$$

Conservative gradient flow

$$f(\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{u}\sim\rho}\phi(\boldsymbol{x},\boldsymbol{u})$$

Example: $\boldsymbol{u} = (a, \boldsymbol{w}), \phi(\boldsymbol{x}, \boldsymbol{u}) = a\sigma(\boldsymbol{w}^T\boldsymbol{x})$

$$V(\boldsymbol{u}) = \frac{\delta\mathcal{R}}{\delta\rho}(\boldsymbol{u}) = \mathbb{E}_{\boldsymbol{x}}[(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))\phi(\boldsymbol{x},\boldsymbol{u})] = \int K(\boldsymbol{u},\tilde{\boldsymbol{u}})\rho(d\tilde{\boldsymbol{u}}) - \tilde{f}(\boldsymbol{u})$$

$$\partial_t\rho = \nabla(\rho\nabla V)$$

- This is the mean-field equation derived by Chizat and Bach (2018), Mei, Montanari and Nguyen (2018), Rotskoff and Vanden-Eijnden (2018), Sirignano and Spiliopoulos (2018), by studying the continuum limit of two-layer neural networks.
- It is the gradient flow of $\mathcal{R}$ under the Wasserstein metric.
- $\mathcal{R}$ is convex but NOT displacement convex.

Discretizing the gradient flows

- Discretizing the population risk (into the empirical risk) using data
- Discretizing the gradient flow
  - particle method – the dynamic version of Monte Carlo
  - smoothes particle method – analog of vortex blob method
  - spectral method – very effective in low dimensions

We will see that gradient descent algorithm (GD) for random feature and neural network models are simply the particle method discretization of the gradient flows discussed before.

Particle method for the feature-based model

$$\partial_t a(\boldsymbol{w}, t) = -\frac{\delta \mathcal{R}}{\delta a}(\boldsymbol{w}) = -\int a(\tilde{\boldsymbol{w}}, t) K(\boldsymbol{w}, \tilde{\boldsymbol{w}}) \pi(d\tilde{\boldsymbol{w}}) + \tilde{f}(\boldsymbol{w})$$

$$\pi(d\boldsymbol{w}) \sim \frac{1}{m} \sum_j \delta_{\boldsymbol{w}_j}, \, a(\boldsymbol{w}_j, t) \sim a_j(t)$$

Discretized version:

$$\frac{d}{dt} a_j(t) = -\frac{1}{m} \sum_k K(\boldsymbol{w}_j, \boldsymbol{w}_k) a_k(t) + \tilde{f}(\boldsymbol{w}_j)$$

This is exactly the GD for the random feature model.

$$f(\boldsymbol{x}) \sim f_m(\boldsymbol{x}) = \frac{1}{m} \sum_j a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x})$$

Discretization of the conservative flow

$$\partial_t \rho = \nabla(\rho \nabla V)$$

$$\rho(da, d\boldsymbol{w}) \sim \frac{1}{m} \sum_j \delta_{(a_j, \boldsymbol{w}_j)}$$

$$I(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m) = \mathcal{R}(f_m), \quad \boldsymbol{u}_j = (a_j, \boldsymbol{w}_j), j = \in [m]$$

where $f_m(\boldsymbol{x}) = \frac{1}{m} \sum_j a_j \sigma(\boldsymbol{w}_j^T \boldsymbol{x})$.
Lemma: Given a set of initial data $\{\boldsymbol{u}_j^0 = (a_j^0, \boldsymbol{w}_j^0), j \in [m]\}$. The solution of (**??**) with initial data $\rho(0) = \frac{1}{m} \sum_{j=1}^m \delta_{\boldsymbol{u}_j^0}$ is given by

$$\rho(t) = \frac{1}{m} \sum_{j=1}^m \delta_{\boldsymbol{u}_j(t)}$$

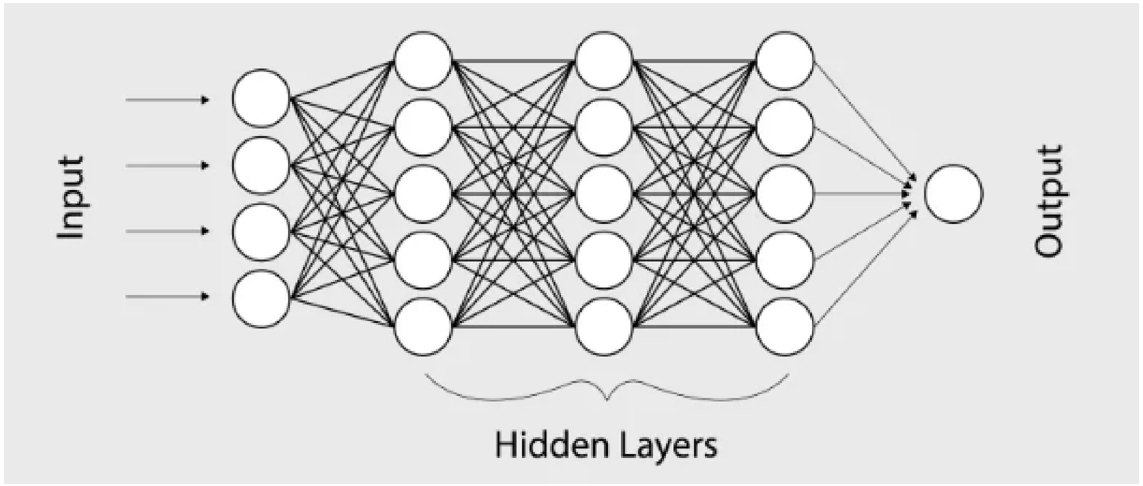
where $\{\boldsymbol{u}_j(\cdot), j = \in [m]\}$ solves the following systems of ODEs:

$$\frac{d\boldsymbol{u}_j}{dt} = -\nabla_{\boldsymbol{u}_j} I(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m), \quad \boldsymbol{u}_j(0) = \boldsymbol{u}_j^0, \quad j \in [m]$$

Note that this is exactly the GD dynamics for two-layer neural networks.

Outline

Deep fully connected neural network models



$$f(\boldsymbol{x}, \theta) = \boldsymbol{W}_L \sigma \circ (\boldsymbol{W}_{L-1} \sigma \circ (\cdots \sigma \circ (\boldsymbol{W}_0 \boldsymbol{x}))), \quad \theta = (\boldsymbol{W}_0, \boldsymbol{W}_1, \cdots, \boldsymbol{W}_L)$$

σ is a scalar function (the activation function), e.g. $\sigma(x) = \max(x, 0)$, ReLU
Numerical instability: Exploding gradients (see Hanin (2018))

$$\nabla_\theta f \sim \boldsymbol{W}_L \cdot \boldsymbol{W}_{L-1} \cdots \boldsymbol{W}_0 \sim \kappa^L, \quad L >> 1$$

Solution: Using residual networks (ResNet, He et al. (2016))

$$\begin{aligned} \boldsymbol{z}_{0,L}(\boldsymbol{x}) &= \boldsymbol{V}\boldsymbol{x}, \\ \boldsymbol{z}_{l+1,L}(\boldsymbol{x}) &= z_{l,L}(\boldsymbol{x}) + \frac{1}{L} \boldsymbol{U}_l \sigma \circ (\boldsymbol{W}_l z_{l,L}(\boldsymbol{x})), \quad l = 0, 1, \cdots, L-1 \\ f(\boldsymbol{x}, \theta) &= \alpha \cdot \boldsymbol{z}_{L,L}(\boldsymbol{x}) \end{aligned}$$

Flow-based representation

Dynamical system viewpoint (E (2017), Haber and Ruthotto (2017) ...)

$$\frac{d\boldsymbol{z}}{d\tau} = \mathbf{g}(\tau, \boldsymbol{z}), \ \boldsymbol{z}(0) = \tilde{\boldsymbol{x}}$$

The flow-map at time 1: $\boldsymbol{x} \rightarrow \boldsymbol{z}(1)$.

Trial functions:

$$f = \alpha^T \boldsymbol{z}(1)$$

Will take $\alpha = \mathbf{1}$ for simplicity.

The correct form of g (E, Ma and Wu, 2019):

$$\mathbf{g}(\tau, \boldsymbol{z}) = \mathbb{E}_{\boldsymbol{w} \sim \pi_\tau} \boldsymbol{a}(\boldsymbol{w}, \tau) \sigma(\boldsymbol{w}^T \boldsymbol{z})$$

where $\{\pi_\tau\}$ is a family of probability distributions.

$$\frac{d\boldsymbol{z}}{d\tau} = \mathbb{E}_{\boldsymbol{w} \sim \pi_\tau} \boldsymbol{a}(\boldsymbol{w}, \tau) \sigma(\boldsymbol{w}^T \boldsymbol{z})$$

As before, we can also use the model:

$$\frac{d\boldsymbol{z}}{d\tau} = \mathbb{E}_{(\boldsymbol{a}, \boldsymbol{w}) \sim \rho_\tau} \boldsymbol{a} \sigma(\boldsymbol{w}^T \boldsymbol{z})$$

$$f(\boldsymbol{x}) = \mathbf{1}^T \boldsymbol{z}_1^{\boldsymbol{x}}$$

Discretize: We obtain the residual neural network model:

$$\boldsymbol{z}_{l+1} = \boldsymbol{z}_l + \frac{1}{LM} \sum_{j=1}^{M} \boldsymbol{a}_{j,l} \sigma(\boldsymbol{z}_l^T \boldsymbol{w}_{j,l}), l = 1, 2, \cdots, L-1, \quad \boldsymbol{z}_0 = V\tilde{\boldsymbol{x}}$$

$$f_L(\boldsymbol{x}) = \mathbf{1}^T \boldsymbol{z}_L$$

Gradient flow for flow-based models

- Consider a generalized flow-based model

$$\begin{aligned} \boldsymbol{z}_0^{\boldsymbol{x}} &= V\boldsymbol{x} \\ \frac{d\boldsymbol{z}_\tau^{\boldsymbol{x}}}{d\tau} &= \mathbb{E}_{\boldsymbol{w}\sim\rho_\tau}[\boldsymbol{\phi}(\boldsymbol{z}_\tau^{\boldsymbol{x}}, \boldsymbol{w})] \\ f(\boldsymbol{x};\rho) &= \mathbf{1}^T\boldsymbol{z}_1^{\boldsymbol{x}}. \end{aligned}$$

- Consider minimizing the following risk

$$\mathcal{R}(f) = \mathbb{E}_{\boldsymbol{x}}(f(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2.$$

- The parameters that need to be optimized is a family of prob distributions:

$$\theta = \{\rho_\tau, \tau \in [0,1]\}$$

- Denote by $W := \{\rho : [0,1] \mapsto \mathcal{P}_2(\Omega)\}$ the space of all feasible parameters. For any $\rho^1, \rho^2 \in W$, define the following metric:

$$d(\rho^1,\rho^2) := \sqrt{\int_0^1 W_2^2(\rho_\tau^1, \rho_\tau^2)d\tau}.$$

Define the Hamiltonian $H : \mathbb{R}^d \times \mathbb{R}^d \times \mathcal{P}_2(\Omega) :\mapsto \mathbb{R}$ as

$$H(\boldsymbol{z}, \boldsymbol{p}, \mu) = \mathbb{E}_{\boldsymbol{w}\sim\mu}[\boldsymbol{p}^T \boldsymbol{\phi}(\boldsymbol{x}, \boldsymbol{w})].$$

Theorem

The gradient flow in the metric space (W, d) is given by

$$\partial_t \rho_\tau(\boldsymbol{w}, t) = \nabla \cdot \left(\rho_\tau(\boldsymbol{w}, t) \nabla V(\boldsymbol{w}; \rho)\right), \ \forall \tau \in [0, 1],$$

where

$$V(\boldsymbol{w}; \rho) = \mathbb{E}_{\boldsymbol{x}}[\frac{\delta H}{\delta \mu}\left(\boldsymbol{z}_\tau^{t,\boldsymbol{x}}, \boldsymbol{p}_\tau^{t,\boldsymbol{x}}, \rho_\tau(\cdot; t)\right)],$$

and for each $\boldsymbol{x}$, $(\boldsymbol{z}_\tau^{t,\boldsymbol{x}}, \boldsymbol{p}_\tau^{t,\boldsymbol{x}})$ are defined by the forward and backward equations, respectively:

$$\frac{d\boldsymbol{z}_\tau^{t,\boldsymbol{x}}}{d\tau} = \nabla_{\boldsymbol{p}} H = \mathbb{E}_{\boldsymbol{w}\sim\rho_\tau(\cdot;t)}[\boldsymbol{\phi}(\boldsymbol{z}_\tau^{t,\boldsymbol{x}}, \boldsymbol{w})]$$
$$\frac{d\boldsymbol{p}_\tau^{t,\boldsymbol{x}}}{d\tau} = -\nabla_{\boldsymbol{z}} H = \mathbb{E}_{\boldsymbol{w}\sim\rho_\tau(\cdot;t)}[\nabla_{\boldsymbol{z}}^T \boldsymbol{\phi}(\boldsymbol{z}_\tau^{t,\boldsymbol{x}}, \boldsymbol{w}) \boldsymbol{p}_\tau^{t,\boldsymbol{x}}].$$

with the boundary conditions:

$$\boldsymbol{z}_0^{t,\boldsymbol{x}} = V\tilde{\boldsymbol{x}}$$
$$\boldsymbol{p}_1^{t,\boldsymbol{x}} = \mathbf{1}2(f(\boldsymbol{x}; \rho(\cdot; t)) - f^*(\boldsymbol{x})).$$

Discretization

- forward Euler for the flow in τ variable, step size $1/L$.
- particle method for the GD dynamics, M samples in each layer

$$
\begin{aligned}
\boldsymbol{z}_{l+1}^{t,\boldsymbol{x}} &= \boldsymbol{z}_l^{t,\boldsymbol{x}} + \frac{1}{LM}\sum_{j=1}^{M} \boldsymbol{\phi}(\boldsymbol{z}_l^{t,\boldsymbol{x}}, \boldsymbol{w}_l^j(t)), \quad l = 0, \dots, L-1 \\
\boldsymbol{p}_l^{t,\boldsymbol{x}} &= \boldsymbol{p}_{l+1}^{t,\boldsymbol{x}} + \frac{1}{LM}\sum_{j=1}^{M} \nabla_{\boldsymbol{z}}\boldsymbol{\phi}(\boldsymbol{z}_{l+1}^{t,\boldsymbol{x}}, \boldsymbol{w}_{l+1}^j(t))\boldsymbol{p}_{l+1}^{t,\boldsymbol{x}}, \quad l = 0, \dots, L-1 \\
\frac{d\boldsymbol{w}_l^j(t)}{dt} &= -\mathbb{E}_{\boldsymbol{x}}[\nabla_{\boldsymbol{w}}^T \boldsymbol{\phi}(\boldsymbol{z}_l^{t,\boldsymbol{x}}, \boldsymbol{w}_l^j(t))\boldsymbol{p}_l^{t,\boldsymbol{x}}].
\end{aligned}
$$

This recovers the gradient descent algorithm (with back-propagation) for the ResNet:

$$
\boldsymbol{z}_{l+1} = \boldsymbol{z}_l + \frac{1}{LM}\sum_{j=1}^{M} \boldsymbol{\phi}(\boldsymbol{z}_l, \boldsymbol{w}_l).
$$

Compositional law of large numbers

Consider the following compositional scheme:

$$\begin{aligned}\boldsymbol{z}_{0,L}(\boldsymbol{x}) &= \boldsymbol{x}, \\ \boldsymbol{z}_{l+1,L}(\boldsymbol{x}) &= z_{l,L}(\boldsymbol{x}) + \frac{1}{LM}\sum_{k=1}^{M} \boldsymbol{a}_{l,k}\sigma(\boldsymbol{w}_{l,k}^T z_{l,L}(\boldsymbol{x})),\end{aligned}$$

$(\boldsymbol{a}_{l,k}, \boldsymbol{w}_{l,k})$ are pairs of vectors i.i.d. sampled from a distribution ρ.

Theorem (E, Ma and Wu 2019)

Assume that

$$\mathbb{E}_\rho \| |\boldsymbol{a}||\boldsymbol{w}^T| \|_F^2 < \infty$$

where for a matrix or vector $\boldsymbol{A}$, $|\boldsymbol{A}|$ means taking element-wise absolute value for $\boldsymbol{A}$. Define $\boldsymbol{z}$ by

$$\begin{aligned}\boldsymbol{z}(\boldsymbol{x}, 0) &= \boldsymbol{V}\boldsymbol{x}, \\ \frac{d}{d\tau}\boldsymbol{z}(\boldsymbol{x}, t) &= \mathbb{E}_{(\boldsymbol{a},\boldsymbol{w})\sim\rho}\boldsymbol{a}\sigma(\boldsymbol{w}^T\boldsymbol{z}(\boldsymbol{x}, \tau)).\end{aligned}$$

Then we have

$$\boldsymbol{z}_{L,L}(\boldsymbol{x}) \to \boldsymbol{z}(\boldsymbol{x}, 1)$$

almost surely as $L \to +\infty$.

Compositional function spaces

Extension: Let $\{\rho_\tau\}$ be a family of prob distributions (for $(\boldsymbol{a}, \boldsymbol{w})$) such that $\mathbb{E}_{\rho_\tau} g(\boldsymbol{a}, \boldsymbol{w})$ is integrable as a function of τ for any continuous function g. Define:

$$\begin{aligned} \boldsymbol{z}(\boldsymbol{x}, 0) &= \boldsymbol{V}\boldsymbol{x}, \\ \frac{d}{d\tau}\boldsymbol{z}(\boldsymbol{x}, \tau) &= \mathbb{E}_{(\boldsymbol{a},\boldsymbol{w})\sim\rho_\tau} \boldsymbol{a}\sigma(\boldsymbol{w}^T \boldsymbol{z}(\boldsymbol{x}, \tau)) \end{aligned}$$

Let $f_{\alpha,\{\rho_\tau\},\boldsymbol{V}}(\boldsymbol{x}) = \alpha^T \boldsymbol{z}(\boldsymbol{x}, 1)$. Define "compositional norm":

$$\begin{aligned} \frac{d}{d\tau}\mathbf{N}(t) &= \mathbb{E}_{\rho_\tau} |\boldsymbol{a}||\boldsymbol{w}|^T \mathbf{N}(\tau), \\ \mathbf{N}(0) &= \mathbf{I} \\ \|f\|_{\mathcal{D}_1} &= \inf_{f = f_{\alpha,\{\rho_\tau\},\boldsymbol{V}}} \|\alpha\|_1 \, \|\mathbf{N}(1)\|_{1,1} \, \|\boldsymbol{V}\|_{1,1}, \end{aligned}$$

$\|\cdot\|_{\mathcal{D}_2}$ is defined similarly.

Barron space and the compositional function space

Theorem

$\mathcal{B} \subset \mathcal{D}_2$. There exists constant $C > 0$, such that

$$\|f\|_{\mathcal{D}_2} \leq \sqrt{d+1}\|f\|_{\mathcal{B}}$$

holds for any $f \in \mathcal{B}$,

Theorem (Direct approximation theorem)

Let $f \in L^2(X) \cap \mathcal{D}_2$. There exists a residue-type neural network $f_L(\cdot; \tilde{\theta})$ of input dimension $d+1$ and depth L such that $\|f_L\|_P \lesssim \|f\|_{c_1}^3$ and

$$\int_X |f(\boldsymbol{x}) - f_L((\boldsymbol{x}); \tilde{\theta})|^2 dx \to 0 \lesssim \frac{\|f\|_{c_2}^2}{L}$$

Furthermore, if $f = f_{\alpha,\{\rho_\tau\},\boldsymbol{V}}$ and ρ_τ is Lipschitz continuous in τ, then

$$\int_X |f(\boldsymbol{x}) - f_L((\boldsymbol{x}); \tilde{\theta})|^2 dx \lesssim \frac{\|f\|_{\mathcal{D}_2}^2}{L}$$

Theorem (Inverse approximation theorem)

Let $f \in L^2(X)$. Assume that there is a sequence of residual networks $\{f_L(\boldsymbol{x})\}_{L=1}^{\infty}$ with increasing depth such that $\|f(\boldsymbol{x}) - f_L(\boldsymbol{x})\| \to 0$ as $L \to \infty$. Assume further that the parameters are (entry-wise) bounded, then there exists α, $\{\rho_\tau\}$ and $\boldsymbol{V}$ such that

$$f(\boldsymbol{x}) = f_{\alpha,\{\rho_\tau\},\boldsymbol{V}}(\boldsymbol{x}).$$

Complexity control

Rademacher complexity bound for path norm

Let $\mathcal{F}_{L,Q} = \{f_L : \|f_L\|_{\mathcal{D}_1} \leq Q\}$. Assume $\boldsymbol{x}_i \in [0,1]^d$. Then, for any data set $S = (\boldsymbol{x}_1, ..., \boldsymbol{x}_n)$, we have

$$\mathrm{Rad}_S(\mathcal{F}_{L,Q}) \leq Q^2 \sqrt{\frac{2\log(2d)}{n}}.$$

Regularized model and a priori estimates

Regularized loss function:

$$\mathcal{L}_{n,\lambda}(\theta) = \hat{\mathcal{R}}_n(\theta) + \lambda(\|\theta\|_{\mathcal{D}_1} + 1)\sqrt{\frac{2\log(2d)}{n}}.$$

Theorem (E, Ma and Wang, 2019)

Assume that $f^\star : [0,1]^d \to [-1,1]$ *such that* $f^* \in \mathcal{D}_2$. *Let*

$$\hat{\theta} = \textit{argmin}\ {}_\theta \mathcal{L}_{n,\lambda}(\theta)$$

then if λ *is larger than some constant, and the depth* L *is sufficiently large, for any* $\delta > 0$, *with probability at least* $1-\delta$,

$$\mathcal{R}(\hat{\theta}) \lesssim \frac{\|f^*\|_{\mathcal{D}_2}^2}{L} + \lambda(\|f^*\|_{\mathcal{D}_1}^2 + 1)\sqrt{\frac{\log(2d)}{n}} + \sqrt{\frac{\log(1/\delta)}{n}}.$$

Outline

Numerical analysis viewpoint of ML

1. Understanding the "physics":
   - "implicit regularization" in certain cases
   - degeneracy to random feature model ("neural tangent kernel")
   - resonance and slow deterioration
   - the frequency principle
   - mean-field scaling
2. Design principles:
   - start from continuous formulation, then discretize
   - Monte Carlo as the benchmark
3. Analysis of ML models and algorithms:
   - New function space hierarchies in high dimension: RKHS, Barron space, compositional function spaces, ...

Open questions

1. Asymptotic (large time) convergence of the gradient flows
2. 3 and more layers......
3. Classification problem (need to define the relevant space of probability measures in high dimensions)
4. Density estimation in high dimension, similar issue + regularization

More generally, high dimensional analysis (function spaces, probability distributions, flows, PDEs, etc)

Papers:

- W. E, C. Ma and L. Wu, "A priori estimates for two layer neural networks", arxiv.org/pdf/1810.06397.pdf, 2018.
- W. E, C. Ma and Q.C. Wang, "A priori estimates of the population risk for residual networks", https://arxiv.org/abs/1903.02154, 2018.
- W. E, C. Ma and L. Wu, "A Comparative Analysis of the Optimization and Generalization Property of Two-layer Neural Network and Random Feature Models Under Gradient Descent Dynamics", arxiv.org/abs/1904.04326, 2019.
- W. E, C. Ma and L. Wu, "Barron spaces and compositional function spaces for neural network models" https://arxiv.org/abs/1906.08039, 2019.
- Weinan E, Chao Ma and Lei Wu, "Machine Learning from a Continuous Viewpoint", arxiv.org/abs/1912.12777, 2019.