On the Convergence of Gradient Descent for Wide Two-Layer Neural Networks

Francis Bach

INRIA - Ecole Normale Supérieure, Paris, France

Joint work with Lénaïc Chizat

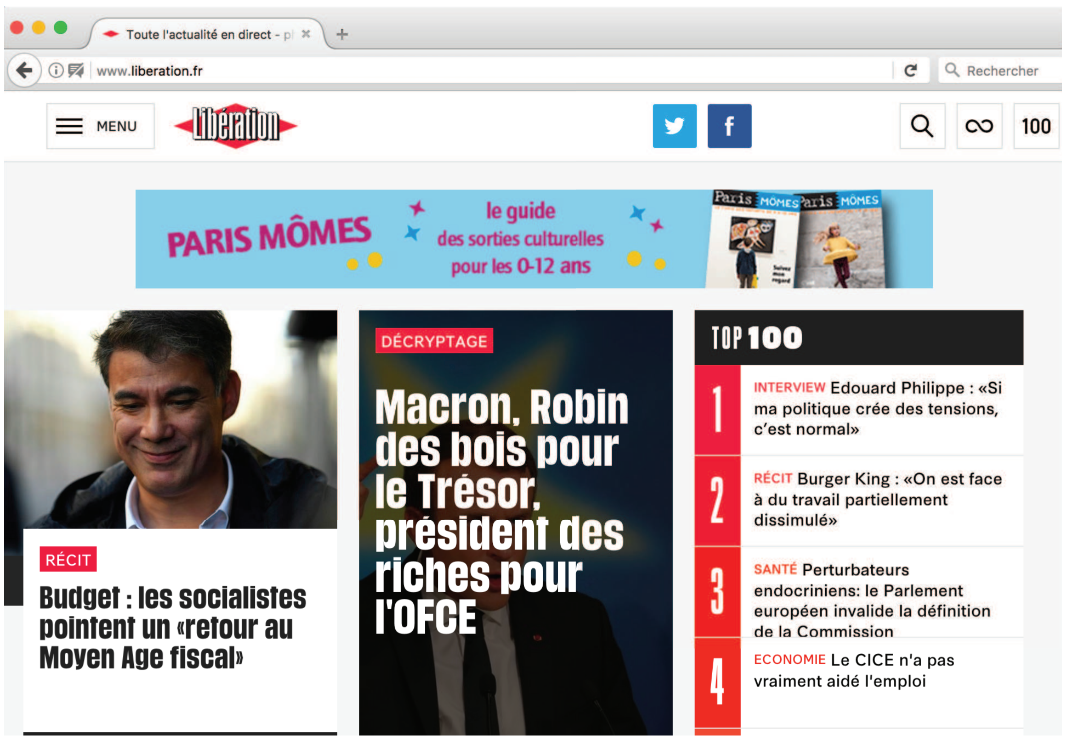
Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \ldots, n$

- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$



- Linear predictions
  - $h(x, \theta) = \theta^\top \Phi(x)$
- E.g., **advertising**: $n > 10^9$
  - $\Phi(x) \in \{0, 1\}^d$, $d > 10^9$
  - Navigation history + ad

Parametric supervised machine learning

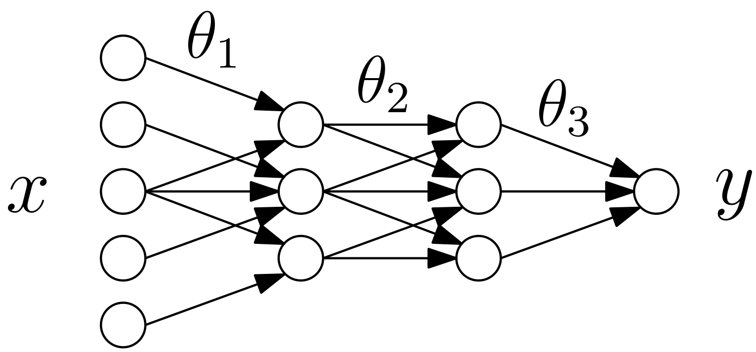
- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$

- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$

x_1 $\quad x_2$ $\quad x_3$ $\quad x_4$ $\quad x_5$ $\quad x_6$

$y_1 = 1 \quad y_2 = 1 \quad y_3 = 1 \quad y_4 = -1 \quad y_5 = -1 \quad y_6 = -1$

  – Neural networks $(n, d > 10^6)$: $h(x, \theta) = \theta_r^\top \sigma(\theta_{r-1}^\top \sigma(\cdots \theta_2^\top \sigma(\theta_1^\top x))$



Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**:

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \quad \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta)$$

data fitting term + regularizer

Parametric supervised machine learning

- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \dots, n$

- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$

- **(regularized) empirical risk minimization**:

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \quad \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta)$$

data fitting term $+$ regularizer

- **Actual goal**: minimize test error $\mathbb{E}_{p(x,y)} \ell(y, h(x, \theta))$

Convex optimization problems

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \ \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta)$$

- **Conditions**: Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

- **Consequences**
  - Efficient algorithms (typically gradient-based)
  - Quantitative runtime and prediction performance guarantees

Convex optimization problems

$$\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n}\sum_{i=1}^{n} \ \ell\big(y_i, h(x_i, \theta)\big) \quad + \quad \lambda \Omega(\theta)$$

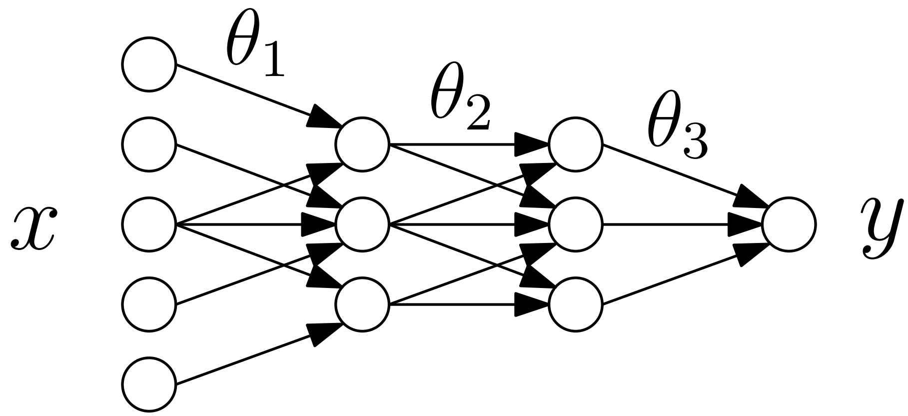
- **Conditions**: Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

- **Consequences**
  - Efficient algorithms (typically gradient-based)
  - Quantitative runtime and prediction performance guarantees

- **Golden years of convexity in machine learning** (1995 to 2020)
  - Support vector machines and kernel methods
  - Sparsity / low-rank models with first-order methods
  - Optimal transport
  - Stochastic methods for large-scale learning and online learning
  - etc.

Theoretical analysis of deep learning

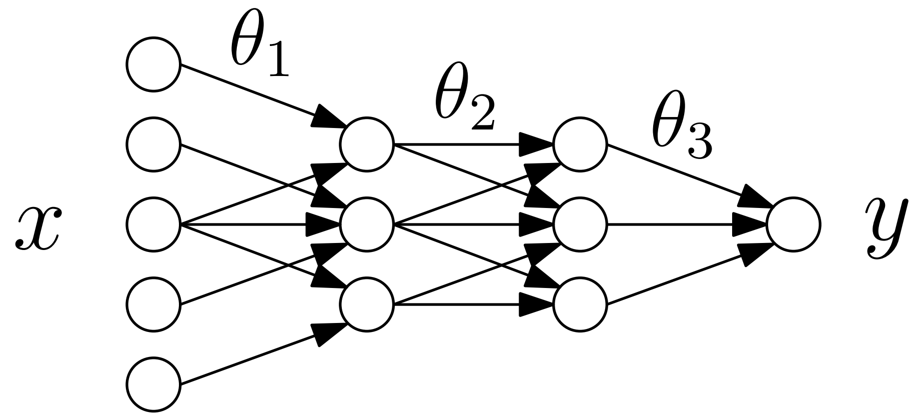
- **Multi-layer neural network** $h(x,\theta) = \theta_r^\top \sigma(\theta_{r-1}^\top \sigma(\cdots \theta_2^\top \sigma(\theta_1^\top x))$



  – NB: already a simplification

Theoretical analysis of deep learning

- **Multi-layer neural network** $h(x,\theta) = \theta_r^\top \sigma(\theta_{r-1}^\top \sigma(\cdots \theta_2^\top \sigma(\theta_1^\top x))$



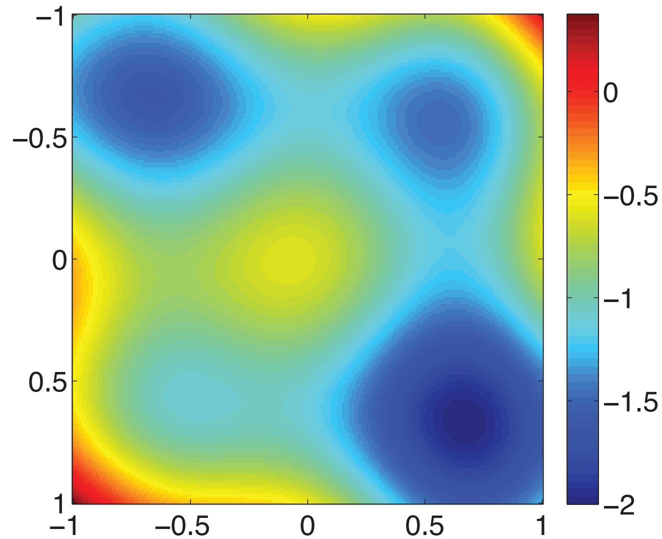
  – NB: already a simplification

- **Main difficulties**

  1. Non-convex optimization problems
  2. Generalization guarantees in the overparameterized regime

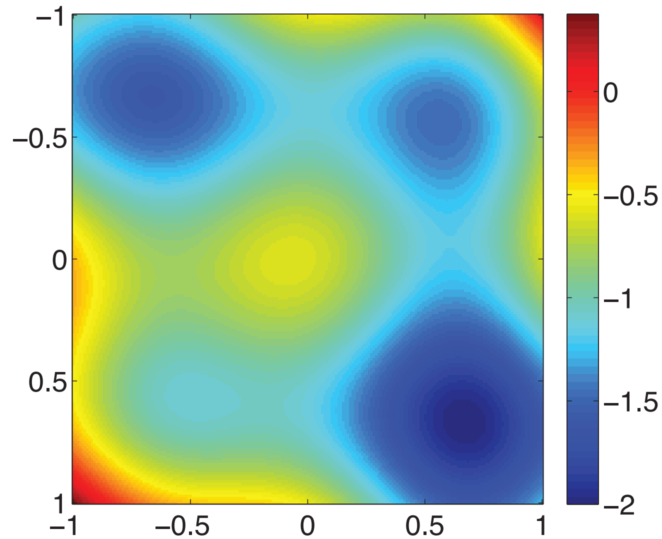
Optimization for multi-layer neural networks

- **What can go wrong with non-convex optimization problems?**
  - Local minima
  - Stationary points
  - Plateaux
  - Bad initialization
  - etc...



Optimization for multi-layer neural networks

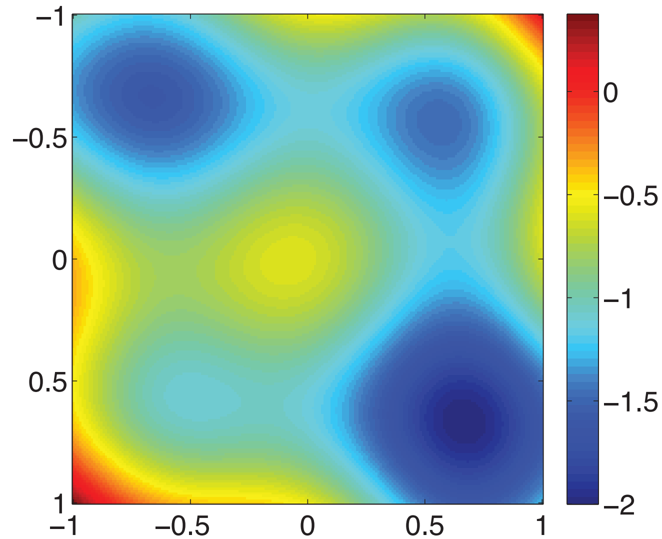
- **What can go wrong with non-convex optimization problems?**
  - Local minima
  - Stationary points
  - Plateaux
  - Bad initialization
  - etc...



- **Generic local theoretical guarantees**
  - Convergence to stationary points or local minima
  - See, e.g., Lee et al. (2016); Jin et al. (2017)

Optimization for multi-layer neural networks

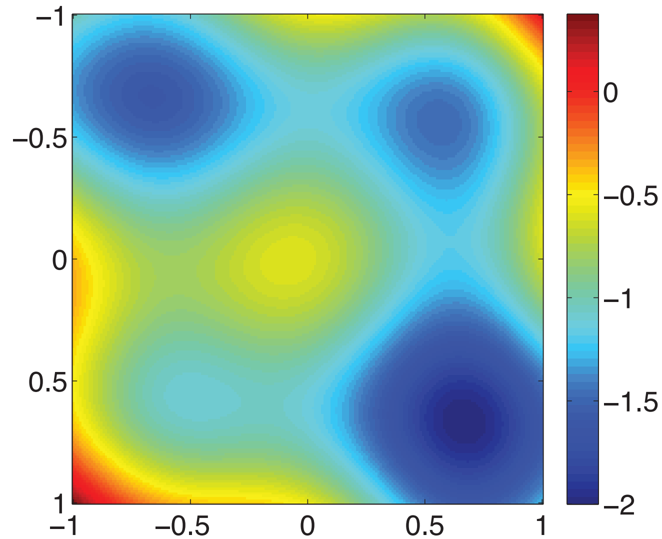
- **What can go wrong with non-convex optimization problems?**
  - Local minima
  - Stationary points
  - Plateaux
  - Bad initialization
  - etc...



- **General global performance guarantees impossible to obtain**

Optimization for multi-layer neural networks

- **What can go wrong with non-convex optimization problems?**
  - Local minima
  - Stationary points
  - Plateaux
  - Bad initialization
  - etc...



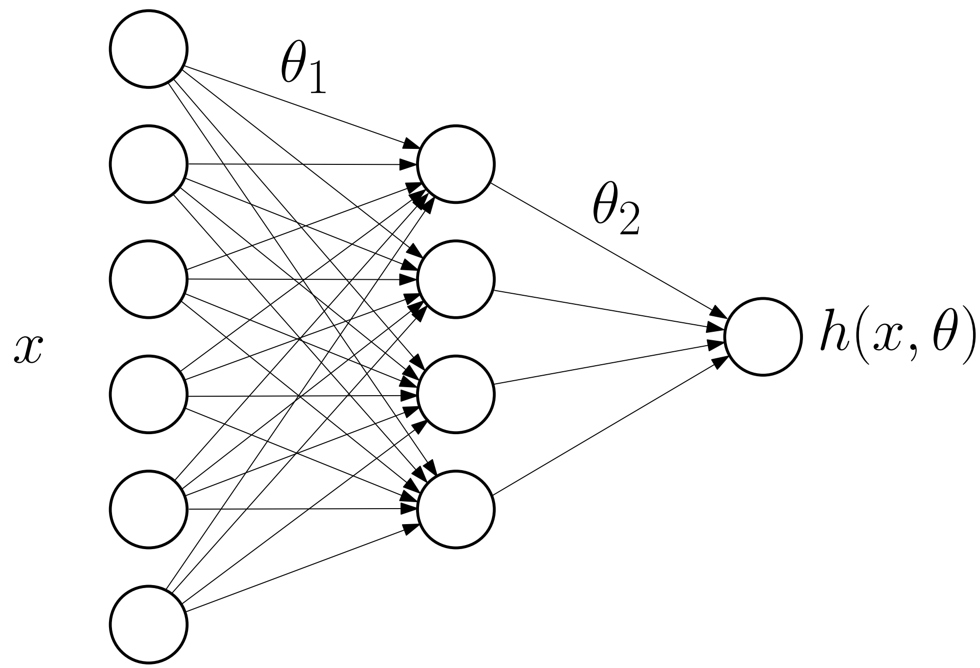
- **General global performance guarantees impossible to obtain**

- **Special case of (deep) neural networks**
  - Most local minima are equivalent (Choromanska et al., 2015)
  - No spurrious local minima (Soltanolkotabi et al., 2018)

Gradient descent for a single hidden layer

- **Predictor**: $h(x) = \frac{1}{m}\theta_2^\top \sigma(\theta_1^\top x) = \frac{1}{m}\sum_{j=1}^m \theta_2(j) \cdot \sigma\big[\theta_1(\cdot, j)^\top x\big]$

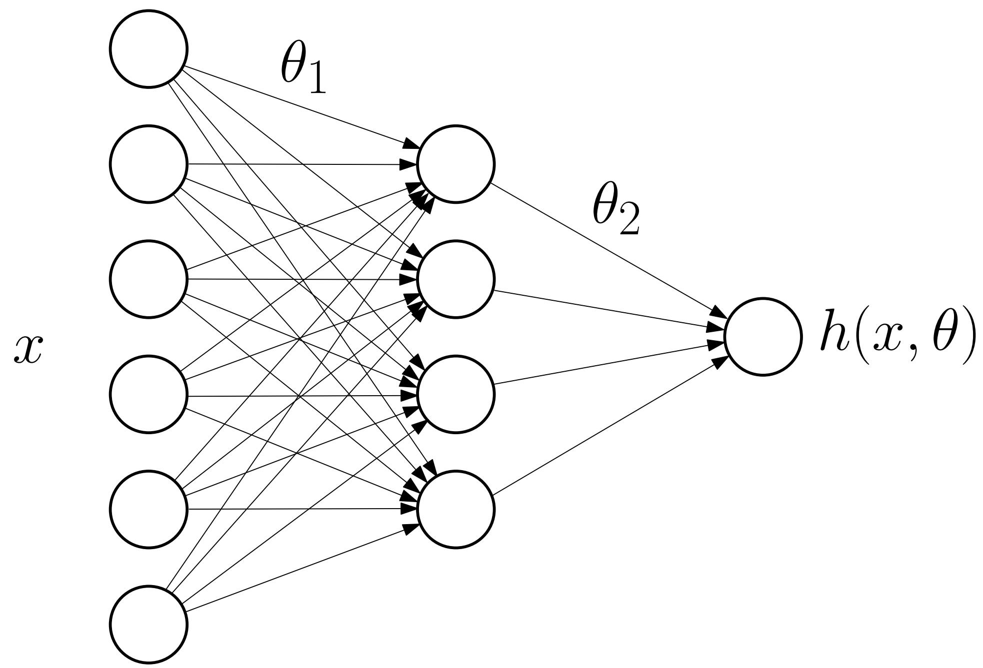
- **Goal**: minimize $R(h) = \mathbb{E}_{p(x,y)} \ell(y, h(x))$, with R convex



Gradient descent for a single hidden layer

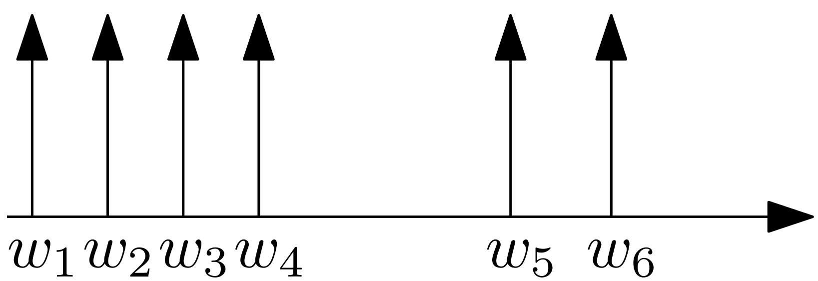
- **Predictor**: $h(x) = \frac{1}{m}\theta_2^\top \sigma(\theta_1^\top x) = \frac{1}{m}\sum_{j=1}^m \theta_2(j) \cdot \sigma\big[\theta_1(\cdot, j)^\top x\big]$
  - Family: $h = \dfrac{1}{m}\displaystyle\sum_{j=1}^m \Psi(w_j)$ with $\Psi(w_j)(x) = \theta_2(j) \cdot \sigma\big[\theta_1(\cdot, j)^\top x\big]$

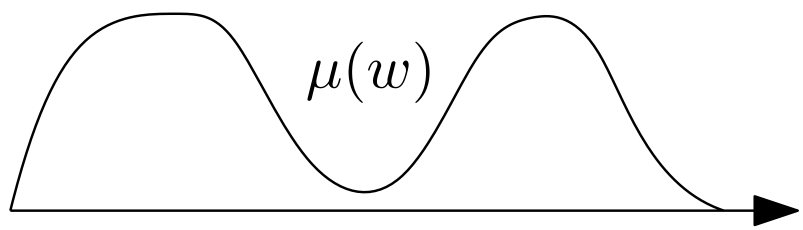
- **Goal**: minimize $R(h) = \mathbb{E}_{p(x,y)}\ell(y, h(x))$, with R convex



Gradient descent for a single hidden layer

- **Predictor**: $h(x) = \frac{1}{m}\theta_2^\top \sigma(\theta_1^\top x) = \frac{1}{m}\sum_{j=1}^m \theta_2(j)\cdot\sigma\big[\theta_1(\cdot,j)^\top x\big]$
  - Family: $h = \frac{1}{m}\sum_{j=1}^m \Psi(w_j)$ with $\Psi(w_j)(x) = \theta_2(j)\cdot\sigma\big[\theta_1(\cdot,j)^\top x\big]$
- **Goal**: minimize $R(h) = \mathbb{E}_{p(x,y)}\ell(y,h(x))$, with R convex
- **Main insight**
  - $h = \frac{1}{m}\sum_{j=1}^m \Psi(w_j) = \int_{\mathcal{W}} \Psi(w)d\mu(w)$ with $d\mu(w) = \frac{1}{m}\sum_{j=1}^m \delta_{w_j}$





Gradient descent for a single hidden layer

- **Predictor**: $h(x) = \frac{1}{m}\theta_2^\top \sigma(\theta_1^\top x) = \frac{1}{m}\sum_{j=1}^m \theta_2(j) \cdot \sigma\big[\theta_1(\cdot,j)^\top x\big]$
  - Family: $h = \displaystyle\frac{1}{m}\sum_{j=1}^m \Psi(w_j)$ with $\Psi(w_j)(x) = \theta_2(j) \cdot \sigma\big[\theta_1(\cdot,j)^\top x\big]$

- **Goal**: minimize $R(h) = \mathbb{E}_{p(x,y)}\ell(y, h(x))$, with R convex

- **Main insight**
  - $h = \displaystyle\frac{1}{m}\sum_{j=1}^m \Psi(w_j) = \int_{\mathcal{W}} \Psi(w) d\mu(w)$ with $d\mu(w) = \displaystyle\frac{1}{m}\sum_{j=1}^m \delta_{w_j}$
  - Overparameterized models with m large $\approx$ measure μ with densities
  - Barron (1993); Kurkova and Sanguineti (2001); Bengio et al. (2006); Rosset et al. (2007); Bach (2017)

Optimization on measures

- **Minimize with respect to measure** μ: $R\Big(\int_{\mathcal{W}} \Psi(w) d\mu(w)\Big)$
  - Convex optimization problem on measures
  - Frank-Wolfe techniques for incremental learning
  - Non-tractable (Bach, 2017), not what is used in practice

Optimization on measures

- **Minimize with respect to measure** μ: $R\Big(\int_{\mathcal{W}} \Psi(w) d\mu(w)\Big)$
  - Convex optimization problem on measures
  - Frank-Wolfe techniques for incremental learning
  - Non-tractable (Bach, 2017), not what is used in practice
- **Represent** μ **by a finite set of "particles"** $\mu = \frac{1}{m}\sum_{j=1}^{m} \delta_{w_j}$
  - Backpropagation = gradient descent on $(w_1, \dots, w_m)$
- **Three questions**:
  - Algorithm limit when number of particles m gets large
  - Global convergence to a global minimizer
  - Prediction performance

Many particle limit and global convergence (Chizat and Bach, 2018)

- **General framework**: minimize $F(\mu) = R\Big(\int_{\mathcal{W}} \Psi(w) d\mu(w) \Big)$
  - Algorithm: minimizing $F_m(w_1, \dots, w_m) = R\Big(\frac{1}{m} \sum_{j=1}^{m} \Psi(w_j) \Big)$

Many particle limit and global convergence (Chizat and Bach, 2018)

- **General framework**: minimize $F(\mu) = R\Big(\int_{\mathcal{W}} \Psi(w) d\mu(w)\Big)$
  - Algorithm: minimizing $F_m(w_1, \dots, w_m) = R\Big(\frac{1}{m}\sum_{j=1}^m \Psi(w_j)\Big)$
  - Gradient flow $\dot{W} = -m\nabla F_m(W)$, with $W = (w_1, \dots, w_m)$
  - Idealization of (stochastic) gradient descent

Many particle limit and global convergence (Chizat and Bach, 2018)

- **General framework**: minimize $F(\mu) = R\Big(\int_{\mathcal{W}} \Psi(w) d\mu(w)\Big)$
  - Algorithm: minimizing $F_m(w_1, \dots, w_m) = R\Big(\frac{1}{m}\sum_{j=1}^{m} \Psi(w_j)\Big)$
  - Gradient flow $\dot{W} = -m \nabla F_m(W)$, with $W = (w_1, \dots, w_m)$
  - Idealization of (stochastic) gradient descent
    1. Single pass SGD on the unobserved expected risk
    2. Multiple pass SGD or full GD on the empirical risk

Many particle limit and global convergence (Chizat and Bach, 2018)

- **General framework**: minimize $F(\mu) = R\Big(\int_{\mathcal{W}} \Psi(w) d\mu(w)\Big)$
  - Algorithm: minimizing $F_m(w_1, \dots, w_m) = R\Big(\frac{1}{m} \sum_{j=1}^{m} \Psi(w_j)\Big)$
  - Gradient flow $\dot{W} = -m \nabla F_m(W)$, with $W = (w_1, \dots, w_m)$
  - Idealization of (stochastic) gradient descent
- **Limit when m tends to infinity**
  - Wasserstein gradient flow (Nitanda and Suzuki, 2017; Chizat and Bach, 2018; Mei, Montanari, and Nguyen, 2018; Sirignano and Spiliopoulos, 2018; Rotskoff and Vanden-Eijnden, 2018)
- NB: for more details on gradient flows, see Ambrosio et al. (2008)

Many particle limit and global convergence (Chizat and Bach, 2018)

- **(informal) theorem**: when the number of particles tends to infinity, the gradient flow converges to the global optimum

Many particle limit and global convergence (Chizat and Bach, 2018)

- **(informal) theorem**: when the number of particles tends to infinity, the gradient flow converges to the global optimum
  - See precise definitions and statement in paper
  - Two key ingredients: homogeneity and initialization

Many particle limit and global convergence (Chizat and Bach, 2018)

- **(informal) theorem**: when the number of particles tends to infinity, the gradient flow converges to the global optimum
  - See precise definitions and statement in paper
  - Two key ingredients: homogeneity and initialization

- **Homogeneity** (see, e.g., Haeffele and Vidal, 2017; Bach et al., 2008)
  - Full or partial, e.g., $\Psi(w_j)(x) = m\theta_2(j) \cdot \sigma\big[\theta_1(\cdot, j)^\top x\big]$
  - Applies to rectified linear units (but also to sigmoid activations)

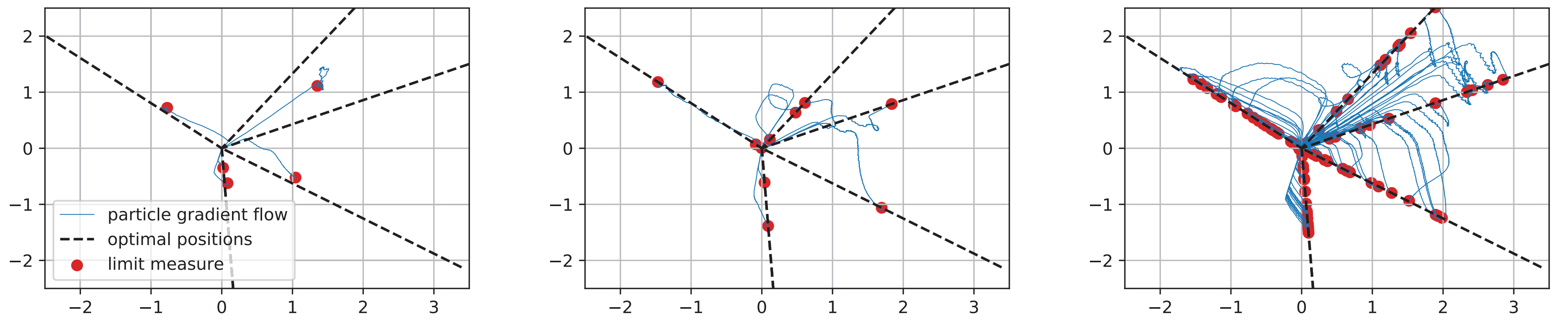
- **Sufficiently spread initial measure**
  - Needs to cover the entire sphere of directions

Many particle limit and global convergence (Chizat and Bach, 2018)

- **(informal) theorem**: when the number of particles tends to infinity, the gradient flow converges to the global optimum
  - See precise definitions and statement in paper
  - Two key ingredients: homogeneity and initialization
- **Homogeneity** (see, e.g., Haeffele and Vidal, 2017; Bach et al., 2008)
  - Full or partial, e.g., $\Psi(w_j)(x) = m\theta_2(j) \cdot \sigma\big[\theta_1(\cdot, j)^\top x\big]$
  - Applies to rectified linear units (but also to sigmoid activations)
- **Sufficiently spread initial measure**
  - Needs to cover the entire sphere of directions
- **Only qualititative!**

Simple simulations with neural networks

- ReLU units with $d = 2$ (optimal predictor has 5 neurons)



5 neurons | 10 neurons | 100 neurons

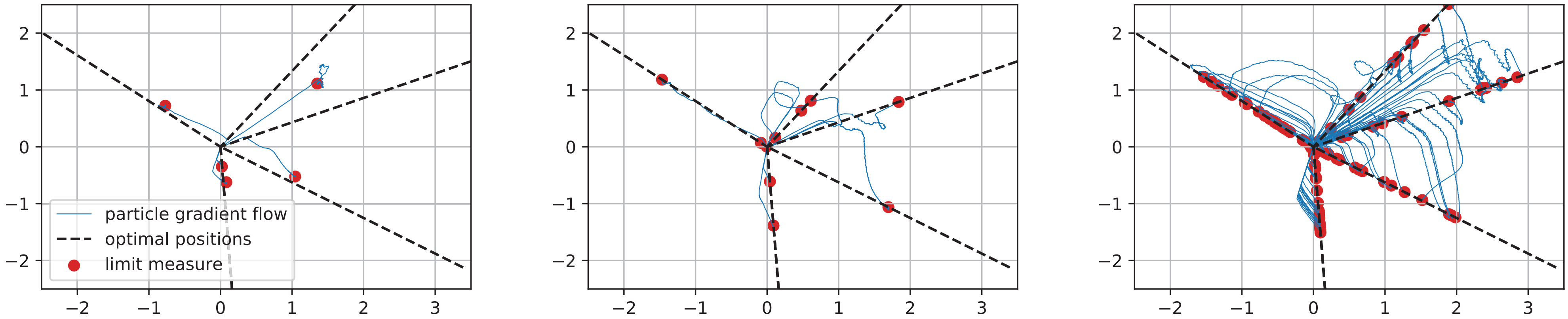
$$h(x) = \frac{1}{m} \sum_{j=1}^{m} \Psi(w_j)(x) = \frac{1}{m} \sum_{j=1}^{m} \theta_2(j) \big(\theta_1(\cdot, j)^\top x\big)_+$$

(plotting $|\theta_2(j)|\theta_1(\cdot, j)$ for each hidden neuron j)

NB : also applies to spike deconvolution

Simple simulations with neural networks

- ReLU units with $d = 2$ (optimal predictor has 5 neurons)



5 neurons 10 neurons 100 neurons

NB : also applies to spike deconvolution

From optimization to statistics

- **Summary**: with $h(x) = \frac{1}{m} \sum_{j=1}^{m} \Psi(w_j)(x) = \frac{1}{m} \sum_{j=1}^{m} \theta_2(j) \big(\theta_1(\cdot, j)^\top x\big)_+$
  - If m tends to infinity, the gradient flow converges to a global minimizer of the risk $R(h) = \mathbb{E}_{p(x,y)} \ell(y, h(x))$
  - Requires well-spread initialization, no quantitative results

From optimization to statistics

- **Summary**: with $h(x) = \frac{1}{m}\sum_{j=1}^{m}\Psi(w_j)(x) = \frac{1}{m}\sum_{j=1}^{m}\theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - If m tends to infinity, the gradient flow converges to a global minimizer of the risk $R(h) = \mathbb{E}_{p(x,y)}\ell(y, h(x))$
  - Requires well-spread initialization, no quantitative results

- **Single-pass SGD** with R the (unobserved) expected risk
  - Converges to an optimal predictor on the testing distribution
  - Tends to underfit

From optimization to statistics

- **Summary**: with $h(x) = \frac{1}{m}\sum_{j=1}^{m} \Psi(w_j)(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - If m tends to infinity, the gradient flow converges to a global minimizer of the risk $R(h) = \mathbb{E}_{p(x,y)}\ell(y, h(x))$
  - Requires well-spread initialization, no quantitative results

- **Single-pass SGD** with R the (unobserved) expected risk
  - Converges to an optimal predictor on the testing distribution
  - Tends to underfit

- **Multiple-pass SGD or full GD** with R the empirical risk
  - Converges to an optimal predictor on the training distribution
  - Should overfit?

Interpolation regime

- Minimizing $R(h) = \frac{1}{n}\sum_{i=1}^{n}\ell(y_i, h(x_i))$ for $h(x) = \frac{1}{m}\sum_{j=1}^{m}\theta_2(j)\big(\theta_1(\cdot, j)^\top x\big)_+$
  - When $m(d+1) > n$, typically there exist many h such that

$$\forall i \in \{1, \dots, n\}, \quad h(x_i) = y_i \qquad \text{(or } \ell(y_i, h(x_i)) = 0\text{)}$$

  - See Belkin et al. (2018); Ma et al. (2018); Vaswani et al. (2019)

Interpolation regime

- Minimizing $R(h) = \frac{1}{n}\sum_{i=1}^{n}\ell(y_i, h(x_i))$ for $h(x) = \frac{1}{m}\sum_{j=1}^{m}\theta_2(j)\big(\theta_1(\cdot, j)^\top x\big)_+$
  - When $m(d+1) > n$, typically there exist many h such that

$$\forall i \in \{1, \dots, n\}, \quad h(x_i) = y_i \qquad \text{(or } \ell(y_i, h(x_i)) = 0\text{)}$$

  - See Belkin et al. (2018); Ma et al. (2018); Vaswani et al. (2019)
- **Which h is the gradient flow converging to?**
  - Implicit bias of (stochastic) gradient descent

Interpolation regime

- Minimizing $R(h) = \frac{1}{n}\sum_{i=1}^{n}\ell(y_i, h(x_i))$ for $h(x) = \frac{1}{m}\sum_{j=1}^{m}\theta_2(j)\big(\theta_1(\cdot, j)^\top x\big)_+$
  - When $m(d+1) > n$, typically there exist many h such that

$$\forall i \in \{1, \dots, n\}, \quad h(x_i) = y_i \qquad \text{(or } \ell(y_i, h(x_i)) = 0\text{)}$$

  - See Belkin et al. (2018); Ma et al. (2018); Vaswani et al. (2019)

- **Which h is the gradient flow converging to?**
  - Implicit bias of (stochastic) gradient descent
  - Typically minimum Euclidean norm solution (Gunasekar et al., 2017; Soudry et al., 2018; Gunasekar et al., 2018)

Interpolation regime

- Minimizing $R(h) = \frac{1}{n}\sum_{i=1}^{n}\ell(y_i, h(x_i))$ for $h(x) = \frac{1}{m}\sum_{j=1}^{m}\theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - When $m(d+1) > n$, typically there exist many h such that

$$\forall i \in \{1,\dots,n\}, \quad h(x_i) = y_i \qquad \text{(or } \ell(y_i, h(x_i)) = 0\text{)}$$

  - See Belkin et al. (2018); Ma et al. (2018); Vaswani et al. (2019)
- **Which h is the gradient flow converging to?**
  - Implicit bias of (stochastic) gradient descent
  - Typically minimum Euclidean norm solution (Gunasekar et al., 2017; Soudry et al., 2018; Gunasekar et al., 2018)
  - Surprisingly difficult for the square loss
  - Surprisingly easy for the logistic loss

Maximum margin and logistic regression

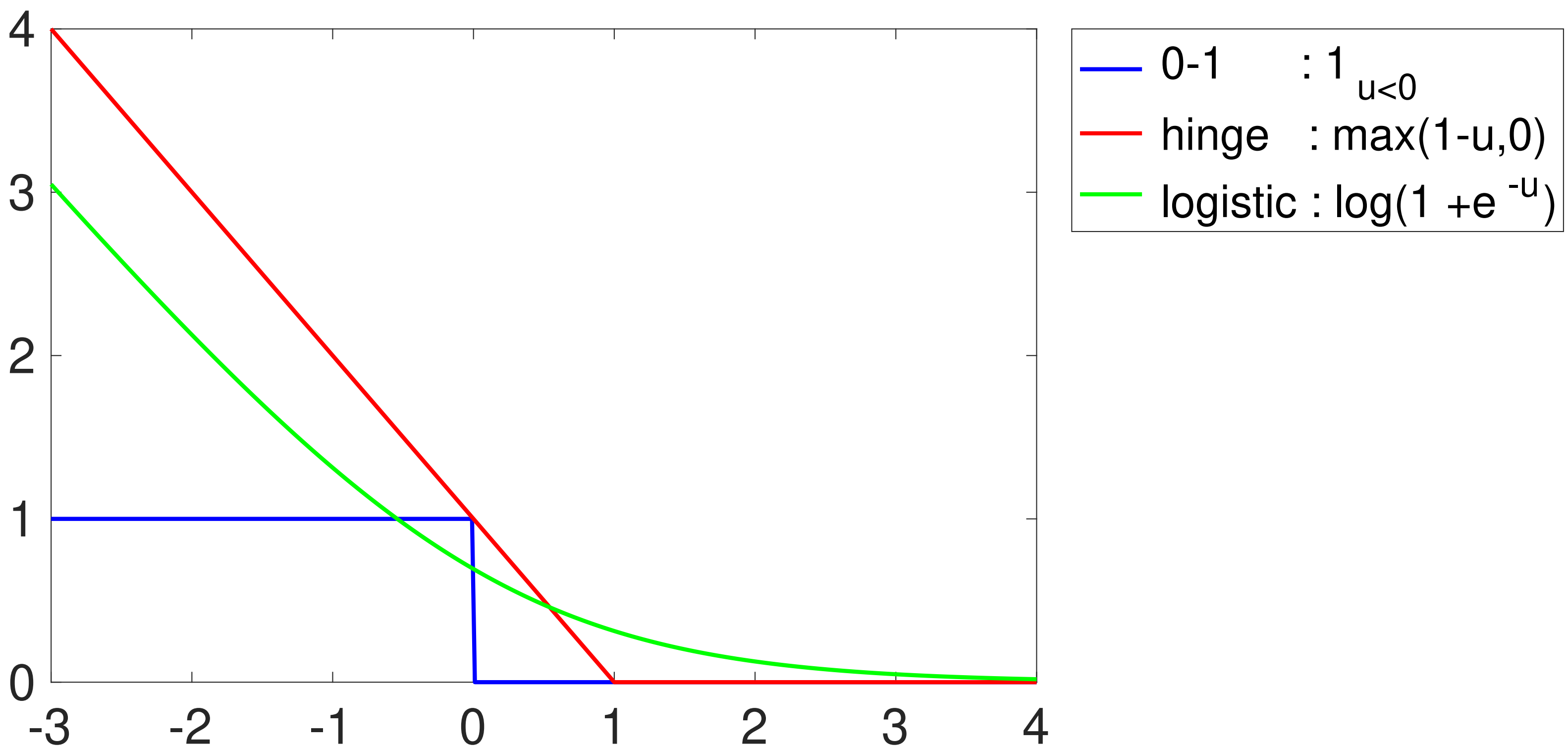
- **Logistic regression:** $\displaystyle\min_{\theta\in\mathbb{R}^d} \ \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-y_i\theta^\top x_i))$
  - Separable data: $\exists\theta\in\mathbb{R}^d, \ \ \forall i\in\{1,\dots,n\}, \ y_i\theta^\top x_i > 0$

Maximum margin and logistic regression

- **Logistic regression:** $\displaystyle \min_{\theta \in \mathbb{R}^d} \ \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\theta^\top x_i))$
  - Separable data: $\exists \theta \in \mathbb{R}^d, \ \ \forall i \in \{1,\dots,n\}, \ y_i\theta^\top x_i > 1$

Maximum margin and logistic regression

- **Logistic regression:** $\min_{\theta \in \mathbb{R}^d} \quad \frac{1}{n} \sum_{i=1}^{n} \log(1 + \exp(-y_i \theta^\top x_i))$
  - Separable data: $\exists \theta \in \mathbb{R}^d, \quad \forall i \in \{1, \dots, n\}, \; y_i \theta^\top x_i > 1$
  - $0 =$ infimum of the risk, attained for infinitely large $\|\theta\|_2$



(with $u = y_i \theta^\top x_i$)

Maximum margin and logistic regression

- **Logistic regression:** $\min_{\theta \in \mathbb{R}^d} \ \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp(-y_i\theta^\top x_i))$
  - Separable data: $\exists \theta \in \mathbb{R}^d, \ \ \forall i \in \{1,\dots,n\}, \ y_i\theta^\top x_i > 1$
  - $0 =$ infimum of the risk, attained for infinitely large $\|\theta\|_2$

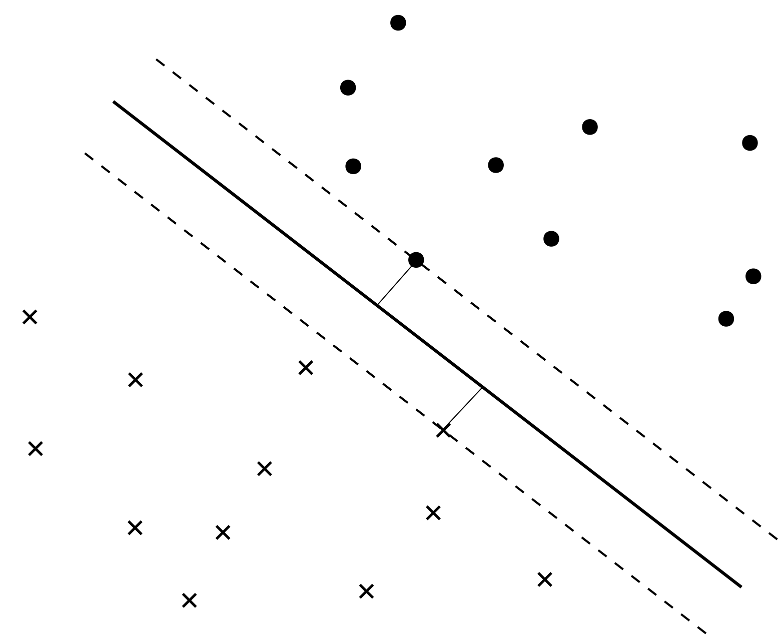
- **Implicit bias of gradient descent** (Soudry et al., 2018)
  - GD diverges but $\frac{1}{\|\theta_t\|_2}\theta_t$ converges to maximum margin separator

$$\max_{\|\eta\|_2=1} \ \min_{i\in\{1,\dots,n\}} y_i \eta^\top x_i$$

  - often written as

$$\min \|\theta\|_2^2 \text{ such that } \forall i, y_i\theta^\top x_i > 1$$

  - Separable support vector machine (Vapnik and Chervonenkis, 1964)



Logistic regression for two-layer neural networks

$$h(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot, j)^\top x\big)_+$$

- **Overparameterized regime** $m \to +\infty$
  - Will converge to well-defined "maximum margin" separator

Logistic regression for two-layer neural networks

$$h(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j) \big(\theta_1(\cdot, j)^\top x\big)_+$$

- **Overparameterized regime** $m \to +\infty$
  - Will converge to well-defined "maximum margin" separator

- **Two different regimes** (Chizat and Bach, 2020)
  1. Optimizing over output layer only θ_2: random feature kernel
  2. Optimizing over all layers θ_1, θ_2: feature learning

Random feature kernel regime - I

- **Prediction function** $h(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - Input weights $\theta_1(\cdot,j)$, $j = 1,\dots,m$, random and fixed
  - Optimize over output weights $\theta_2 \in \mathbb{R}^m$
  - Corresponds to linear predictor with $\Phi(x)_j = \frac{1}{\sqrt{m}}\big(\theta_1(\cdot,j)^\top x\big)_+$

Random feature kernel regime - I

- **Prediction function** $h(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - Input weights $\theta_1(\cdot,j)$, $j = 1,\dots,m$, random and fixed
  - Optimize over output weights $\theta_2 \in \mathbb{R}^m$
  - Corresponds to linear predictor with $\Phi(x)_j = \frac{1}{\sqrt{m}}\big(\theta_1(\cdot,j)^\top x\big)_+$

- **Converges to separator with minimum norm** $\|\theta_2\|_2^2$
  - Direct application of results from Soudry et al. (2018)
  - Limit when m tends to infinity?

Random feature kernel regime - I

- **Prediction function** $h(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - Input weights $\theta_1(\cdot,j)$, $j=1,\dots,m$, random and fixed
  - Optimize over output weights $\theta_2 \in \mathbb{R}^m$
  - Corresponds to linear predictor with $\Phi(x)_j = \frac{1}{\sqrt{m}}\big(\theta_1(\cdot,j)^\top x\big)_+$

- **Converges to separator with minimum norm** $\|\theta_2\|_2^2$
  - Direct application of results from Soudry et al. (2018)
  - Limit when m tends to infinity?

- **Kernel** $\Phi(x)^\top \Phi(x') = \frac{1}{m}\sum_{j=1}^{m} \big(\theta_1(\cdot,j)^\top x\big)_+ \big(\theta_1(\cdot,j)^\top x'\big)_+$
  - Converges to $\mathbb{E}_\eta (\eta^\top x)_+ (\eta^\top x')_+$
  - "Random features" (Neal, 1995; Rahimi and Recht, 2007)

Random feature kernel regime - II

- **Limiting kernel** $\mathbb{E}_\eta (\eta^\top x)_+ (\eta^\top x')_+$
  - Reproducing kernel Hilbert spaces (RKHS) (see, e.g., Schölkopf and Smola, 2001)
  - Space of (very) smooth functions (Bach, 2017)

Random feature kernel regime - II

- **Limiting kernel** $\mathbb{E}_\eta (\eta^\top x)_+ (\eta^\top x')_+$
  - Reproducing kernel Hilbert spaces (RKHS) (see, e.g., Schölkopf and Smola, 2001)
  - Space of (very) smooth functions (Bach, 2017)

- **(informal) theorem** (Chizat and Bach, 2020): when $m \to +\infty$, the gradient flow converges to the function in the RKHS that separates the data with minimum RKHS norm
  - Quantitative analysis available
  - Letting $m \to +\infty$ is useless in practice
  - See Montanari et al. (2019) for related work in the context of "double descent"

From RKHS norm to variation norm

- **Alternative definition of the RKHS norm**

$$\|f\|^2 = \inf_{a(\cdot)} \int_{\mathbb{S}^d} |a(\eta)|^2 d\tau(\eta) \;\; \text{such that} \;\; f(x) = \int_{\mathbb{S}^d} (\eta^\top x)_+ a(\eta) d\tau(\eta)$$

  – Input weigths uniformly distributed on the sphere (Bach, 2017)
  – Smooth functions (does not allow single hidden neuron)

From RKHS norm to variation norm

- **Alternative definition of the RKHS norm**

$$\|f\|^2 = \inf_{a(\cdot)} \int_{\mathbb{S}^d} |a(\eta)|^2 d\tau(\eta) \text{ such that } f(x) = \int_{\mathbb{S}^d} (\eta^\top x)_+ a(\eta) d\tau(\eta)$$

  – Input weigths uniformly distributed on the sphere (Bach, 2017)
  – Smooth functions (does not allow single hidden neuron)

- **Variation norm** (Kurkova and Sanguineti, 2001)

$$\Omega(f) = \inf_{a(\cdot)} \int_{\mathbb{S}^d} |a(\eta)| d\tau(\eta) \text{ such that } f(x) = \int_{\mathbb{S}^d} (\eta^\top x)_+ a(\eta) d\tau(\eta)$$

  – Larger space including non-smooth functions
  – Allows single hidden neuron
  – Adaptivity to linear structures (Bach, 2017)

Feature learning regime

- **Prediction function** $h(x) = \frac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot, j)^\top x\big)_+$
  - Optimize over all weights θ_1, θ_2

Feature learning regime

- **Prediction function** $h(x) = \dfrac{1}{m}\sum_{j=1}^{m} \theta_2(j)\big(\theta_1(\cdot,j)^\top x\big)_+$
  - Optimize over all weights θ_1, θ_2

- **(informal) theorem** (Chizat and Bach, 2020): when $m \to +\infty$, the gradient flow converges to the function that separates the data with minimum **variation norm**
  - Actual learning of representations
  - Adaptivity to linear structures (see Chizat and Bach, 2020)
  - No known convex optimization algorithms in polynomial time
  - End of the curve of double descent (Belkin et al., 2018)

Optimizing over two layers

- Two-dimensional classification with "bias" term

Space of parameters

- Plot of $|\theta_2(j)|\theta_1(\cdot, j)$
- Color depends on sign of $\theta_2(j)$
- "tanh" radial scale

Space of predictors

- $(+/-)$ training set
- One color per class
- Line shows 0 level set of h

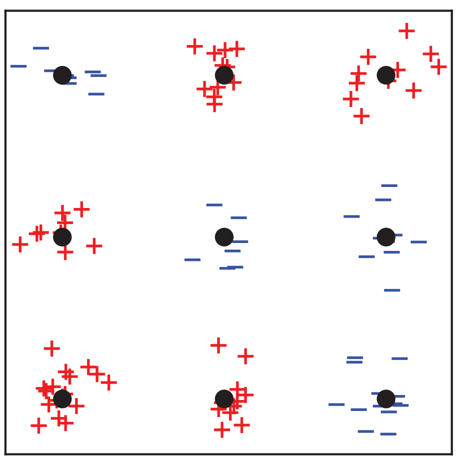
Comparison of kernel and feature learning regimes

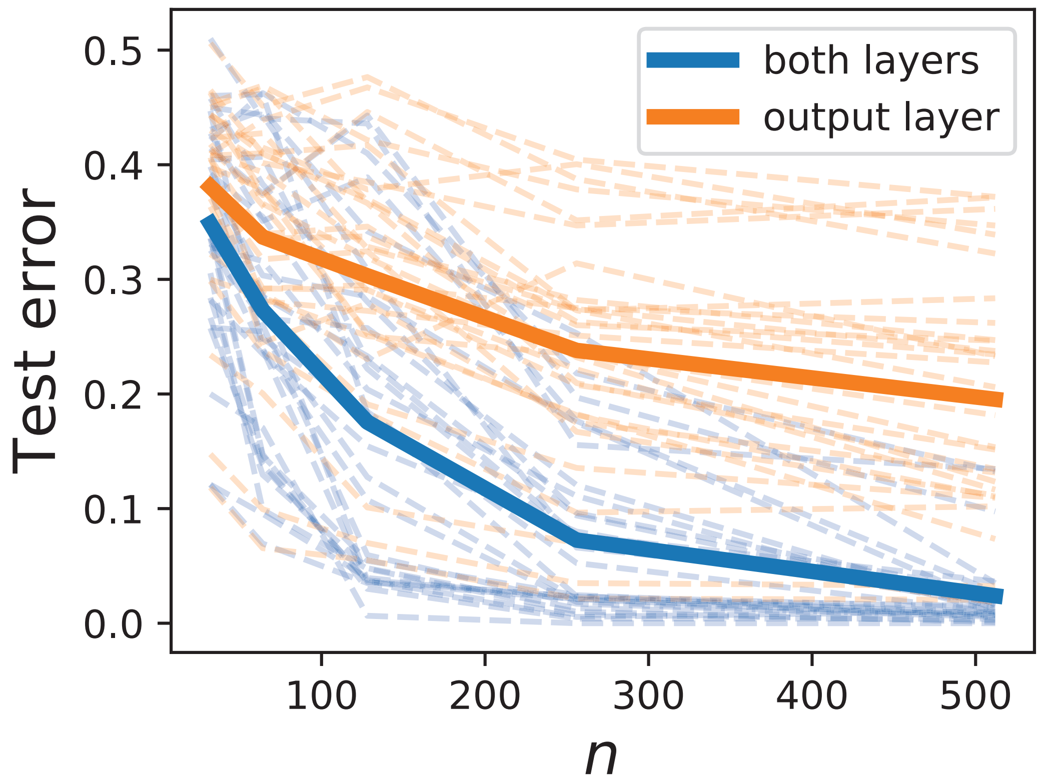
- ℓ_2 (left: kernel) vs. ℓ_1 (right: feature learning and variation norm)

Comparison of kernel and feature learning regimes

- **Adaptivity to linear structures**

- **Two-class classification in dimension** $d = 15$
  - Two first coordinates as shown below
  - All other coordinates uniformly at random





Conclusion

- **Summary**
  - Qualitative analysis of gradient descent for 2-layer neural networks
  - Global convergence with infinitely many neurons
  - Convergence to maximum margin separators in well-defined function spaces
  - Only qualitative

Conclusion

- **Summary**
  - Qualitative analysis of gradient descent for 2-layer neural networks
  - Global convergence with infinitely many neurons
  - Convergence to maximum margin separators in well-defined function spaces
  - Only qualitative

- **Open problems**
  - Quantitative analysis in terms of number of neurons m and time t
  - Extension to convolutional neural networks
  - Extension to deep neural networks

References

Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media, 2008.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(1):629–681, 2017.

Francis Bach, Julien Mairal, and Jean Ponce. Convex sparse matrix factorizations. Technical Report 0812.1869, arXiv, 2008.

A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine learning and the bias-variance trade-off. *arXiv preprint arXiv:1812.11118*, 2018.

Y. Bengio, N. Le Roux, P. Vincent, O. Delalleau, and P. Marcotte. Convex neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

Lénaïc Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, pages 3036–3046, 2018.

Lénaïc Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. *arXiv preprint arXiv:2002.04486*, 2020.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.

Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pages 6151–6159, 2017.

Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pages 1832–1841, 2018.

Benjamin D. Haeffele and René Vidal. Global optimality in neural network training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7331–7339, 2017.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M. Kakade, and Michael I. Jordan. How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887*, 2017.

V. Kurkova and M. Sanguineti. Bounds on rates of variable-basis and neural-network approximation. *IEEE Transactions on Information Theory*, 47(6):2659–2665, Sep 2001.

Jason D. Lee, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. Gradient descent only converges to minimizers. In *Conference on Learning Theory*, pages 1246–1257, 2016.

Siyuan Ma, Raef Bassily, and Mikhail Belkin. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. In *International Conference on Machine Learning*, pages 3331–3340, 2018.

Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layers neural networks. Technical Report 1804.06561, arXiv, 2018.

Andrea Montanari, Feng Ruan, Youngtak Sohn, and Jun Yan. The generalization error of max-margin linear classifiers: High-dimensional asymptotics in the overparametrized regime. *arXiv preprint* *arXiv:1911.01544*, 2019.

R. M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

Atsushi Nitanda and Taiji Suzuki. Stochastic particle gradient descent for infinite ensembles. *arXiv preprint arXiv:1712.05438*, 2017.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20:1177–1184, 2007.

S. Rosset, G. Swirszcz, N. Srebro, and J. Zhu. ℓ_1-regularization in infinite dimensional feature spaces. In *Proceedings of the Conference on Learning Theory (COLT)*, 2007.

Grant M. Rotskoff and Eric Vanden-Eijnden. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. *arXiv preprint arXiv:1805.00915*, 2018.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2001.

Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks. *arXiv preprint arXiv:1805.01053*, 2018.

Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 2018.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1): 2822–2878, 2018.

V. N. Vapnik and A. Ya. Chervonenkis. On a perceptron class. *Avtomat. i Telemekh.*, 25(1):112–120, 1964.

Sharan Vaswani, Francis Bach, and Mark Schmidt. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. In *International Conference on Artificial Intelligence and Statistics*, 2019.