Fundamentals of Machine Learning for Distributed Systems

Francis Bach

INRIA - Ecole Normale Supérieure, Paris, France

Summer school on distributed learning, Sept. 2023

Slides available at www.di.ens.fr/~fbach/rsd2023.pdf

Fundamentals of Machine Learning for Distributed Systems

Francis Bach

INRIA - Ecole Normale Supérieure, Paris, France

Summer school on distributed learning, Sept. 2023

Slides available at www.di.ens.fr/~fbach/rsd2023.pdf

Scientific context

- **Proliferation of digital data**
  - Personal data
  - Industry
  - Scientific: from bioinformatics to humanities
- **Need for automated processing of massive data**

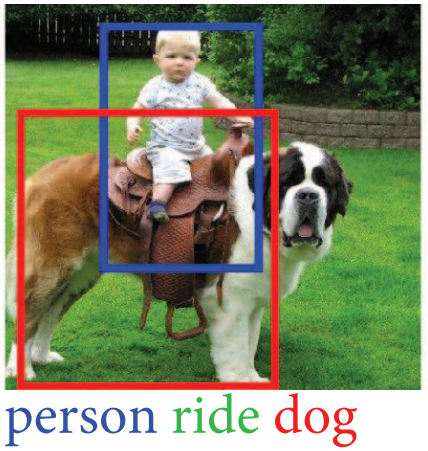
Scientific context

- **Proliferation of digital data**
  - Personal data
  - Industry
  - Scientific: from bioinformatics to humanities
- **Need for automated processing of massive data**
- **Series of "hypes"**

  Big data $\rightarrow$ Data science $\rightarrow$ Machine Learning
  $\rightarrow$ Deep Learning $\rightarrow$ Artificial Intelligence

Recent progress in perception (vision, audio, text)

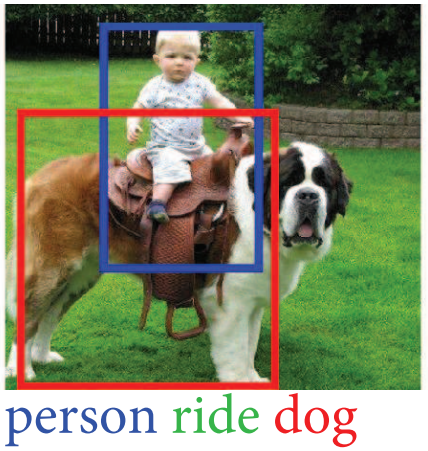
From translate.google.fr



From Peyré et al. (2017)

Recent progress in perception (vision, audio, text)

From translate.google.fr



From Peyré et al. (2017)

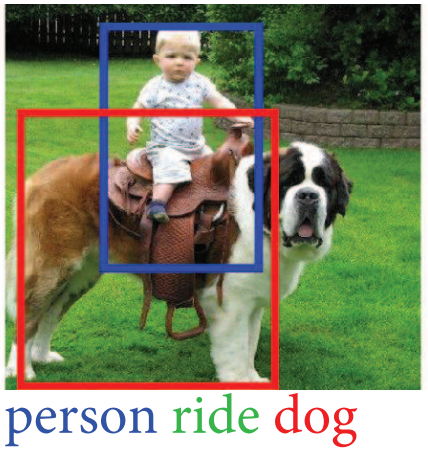
(1) **Massive data**

(2) **Computing power**

(3) **Methodological and scientific progress**

Recent progress in perception (vision, audio, text)

From translate.google.fr



From Peyré et al. (2017)

(1) **Massive data**

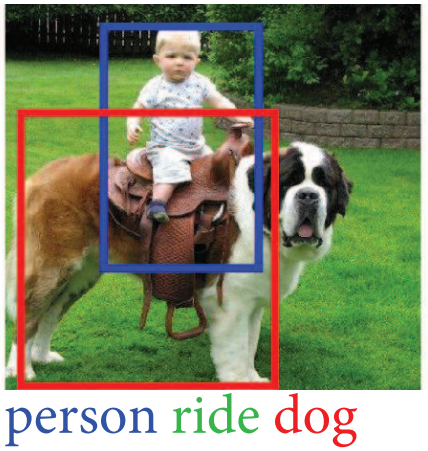
(2) **Computing power**

(3) **Methodological and scientific progress**

"Intelligence" = models + algorithms + data
+ computing power

Recent progress in perception (vision, audio, text)

From translate.google.fr



From Peyré et al. (2017)

(1) **Massive data**

(2) **Computing power**

(3) **Methodological and scientific progress**

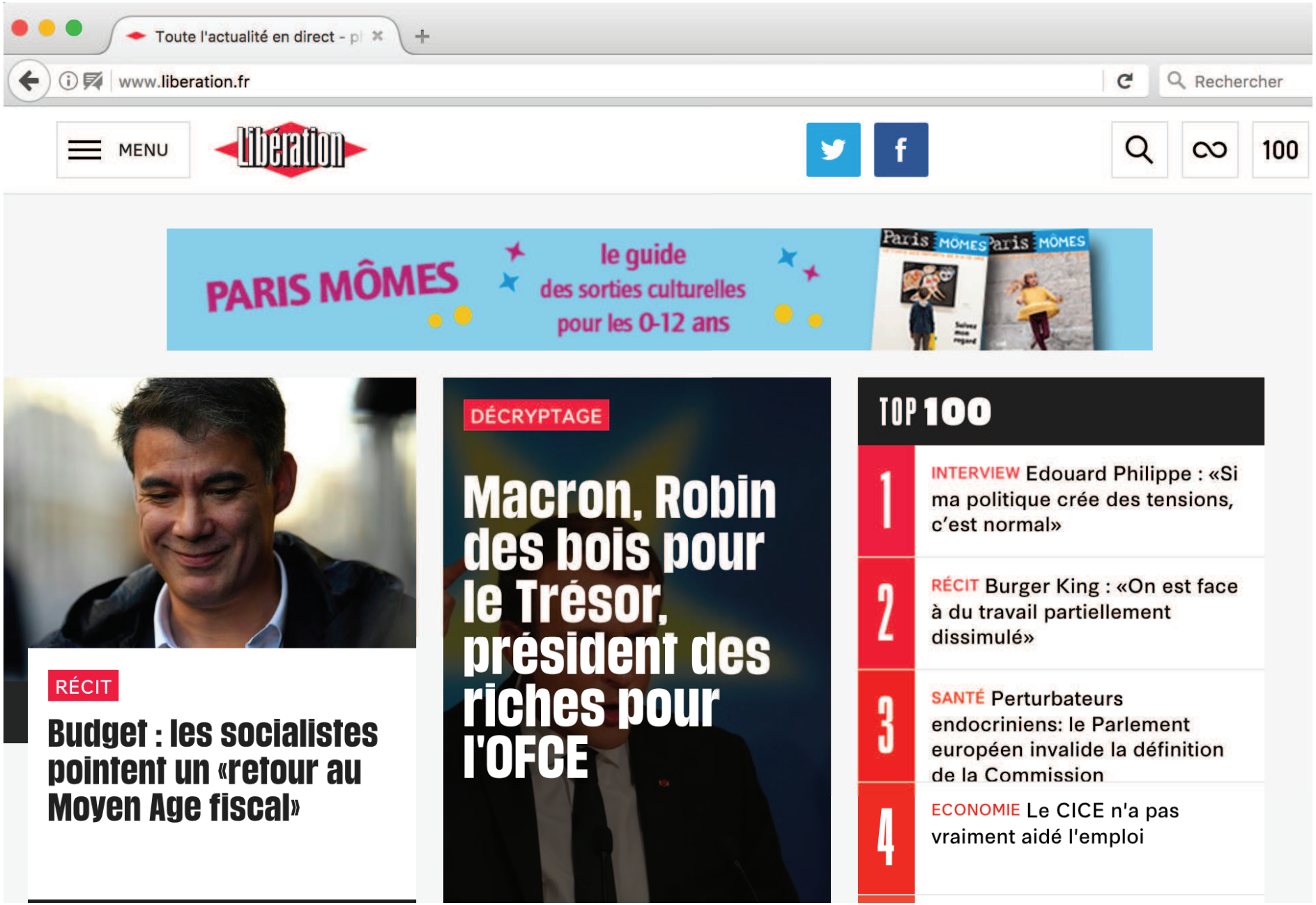
"Intelligence" = models + algorithms + data
+ computing power

Machine learning for large-scale data

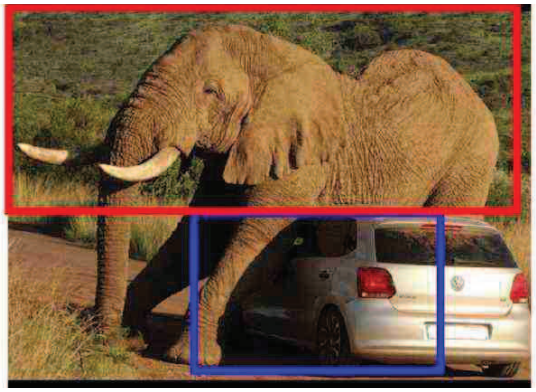
- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input) or number of parameters
  - n : number of observations

- **Examples**: computer vision, advertising, bioinformatics, etc.

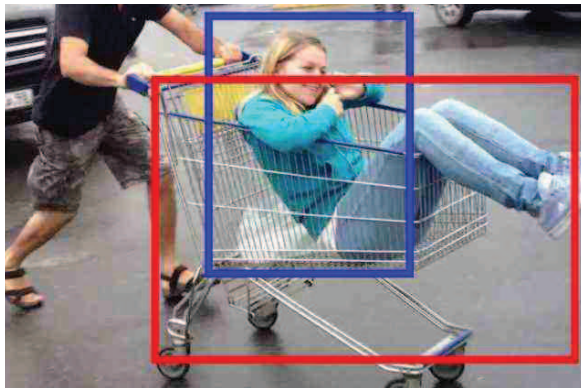
Advertising



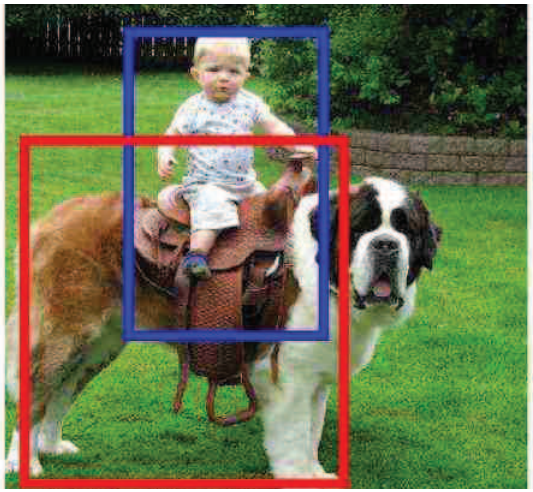
Object / action recognition in images



car under elephant



person in cart

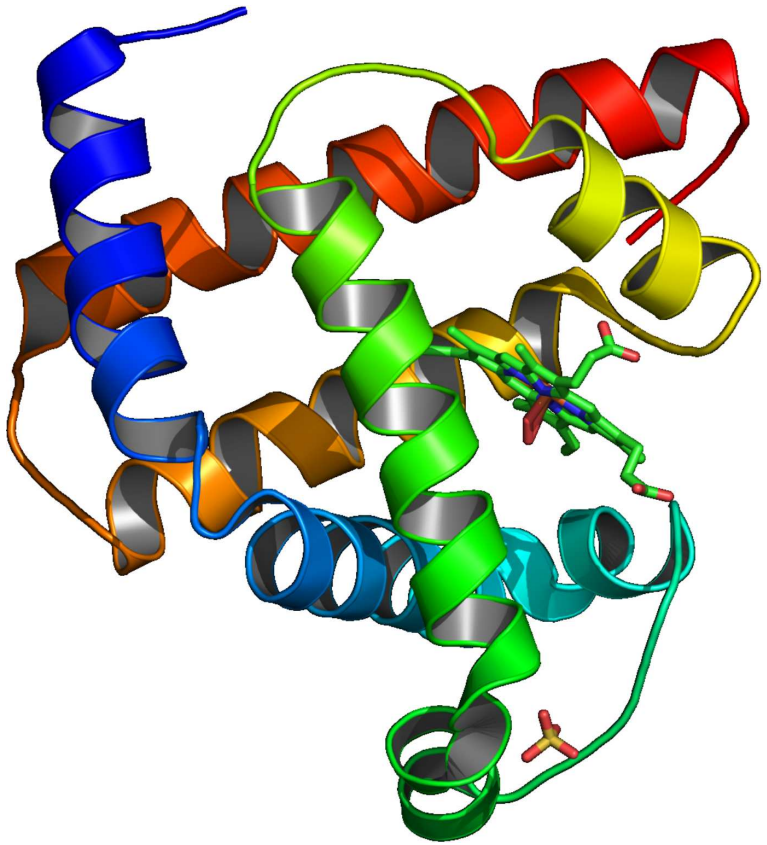


person ride dog

person on top of traffic light

From Peyré, Laptev, Schmid and Sivic (2017)

Bioinformatics



- Predicting multiple functions and interactions of **proteins**
- **Massive data**: up to 1 millions for humans!
- **Complex data**
  - Amino-acid sequence
  - Link with DNA
  - Tri-dimensional molecule

Machine learning for large-scale data

- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input), or number of parameters
  - n : number of observations
- **Examples**: computer vision, advertising, bioinformatics, etc.
- **Ideal running-time complexity**: $O(dn)$ (single machine)

Machine learning for large-scale data

- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input), or number of parameters
  - n : number of observations
- **Examples**: computer vision, advertising, bioinformatics, etc.
- **Ideal running-time complexity**: $O(dn)$ (single machine)
- **Going back to simple methods**
  - Stochastic gradient methods (Robbins and Monro, 1951)
- **Goal: Present classical algorithms and some recent progress**

Machine learning for large-scale data

- **Large-scale supervised machine learning**: **large d, large n**
  - d : dimension of each observation (input), or number of parameters
  - n : number of observations
- **Examples**: computer vision, advertising, bioinformatics, etc.
- **Ideal running-time complexity**: $O(dn)$ (single machine)
- **Going back to simple methods**
  - Stochastic gradient methods (Robbins and Monro, 1951)
- **Goal: Present classical algorithms and some recent progress**
  - *Disclaimer: Significant focus on optimization*

Outline

1. **Introduction/motivation: Supervised machine learning**
   - Machine learning $\approx$ optimization of finite sums
   - Batch optimization methods

2. **Fast stochastic gradient methods for convex problems**
   - Variance reduction: for *training* error
   - Single pass SGD: for *testing* error

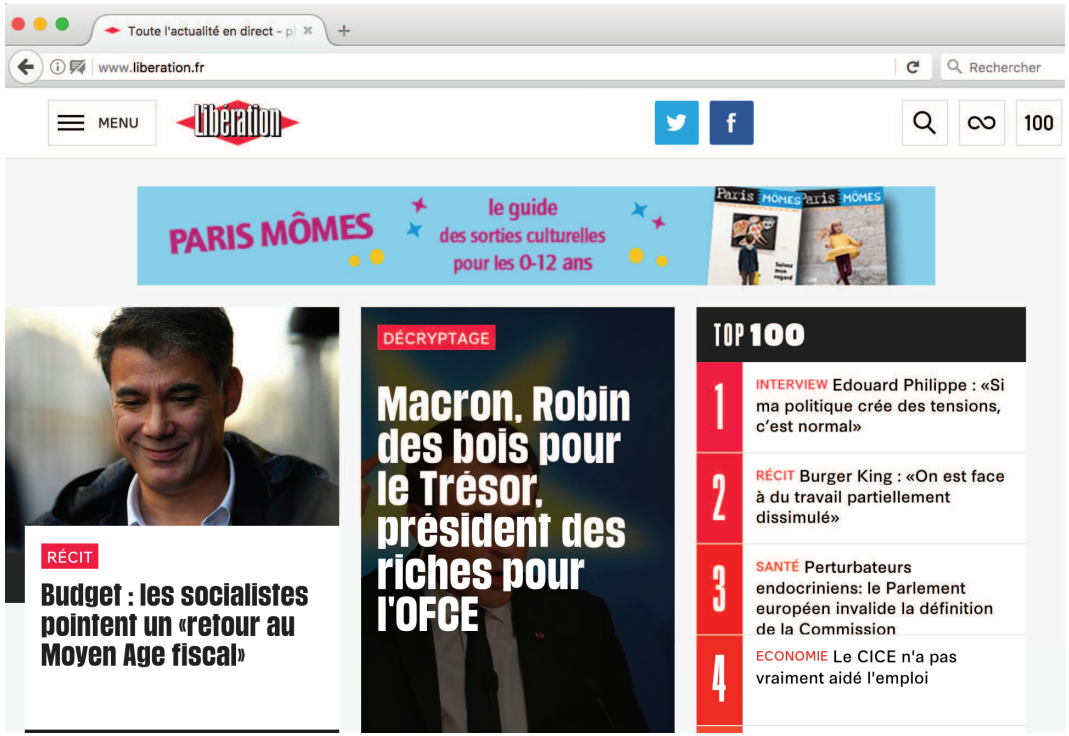
3. **Beyond convex problems**
   - Generic algorithms with generic "guarantees"
   - Global convergence for over-parameterized neural networks

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

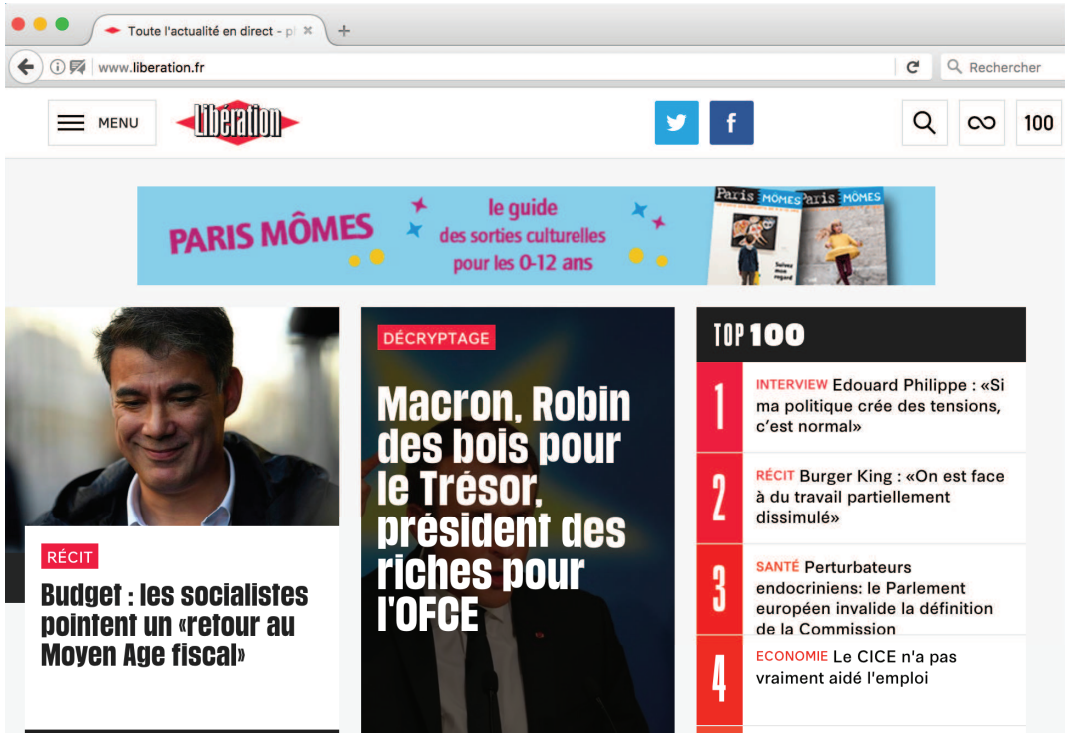
- **Data**: n observations $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$, $i = 1, \ldots, n$
- **Prediction function** $h(x, \theta) \in \mathbb{R}$ parameterized by $\theta \in \mathbb{R}^d$



- **Advertising**: $n > 10^9$
  - $\Phi(x) \in \{0, 1\}^d$, $d > 10^9$
  - Navigation history + ad

Parametric supervised machine learning

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



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

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 $\theta \in \mathbb{R}^d$

x_1 x_2 x_3 x_4 x_5 x_6

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

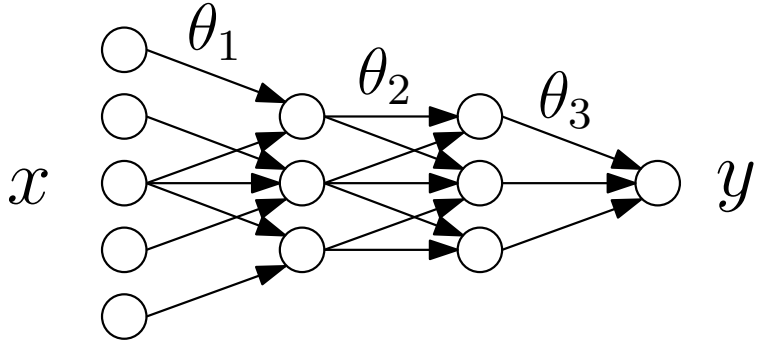
Parametric supervised machine learning

- **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 $\theta \in \mathbb{R}^d$

x_1 x_2 x_3 x_4 x_5 x_6

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

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



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**: find $\hat{\theta}$ solution of

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

data fitting term + regularizer

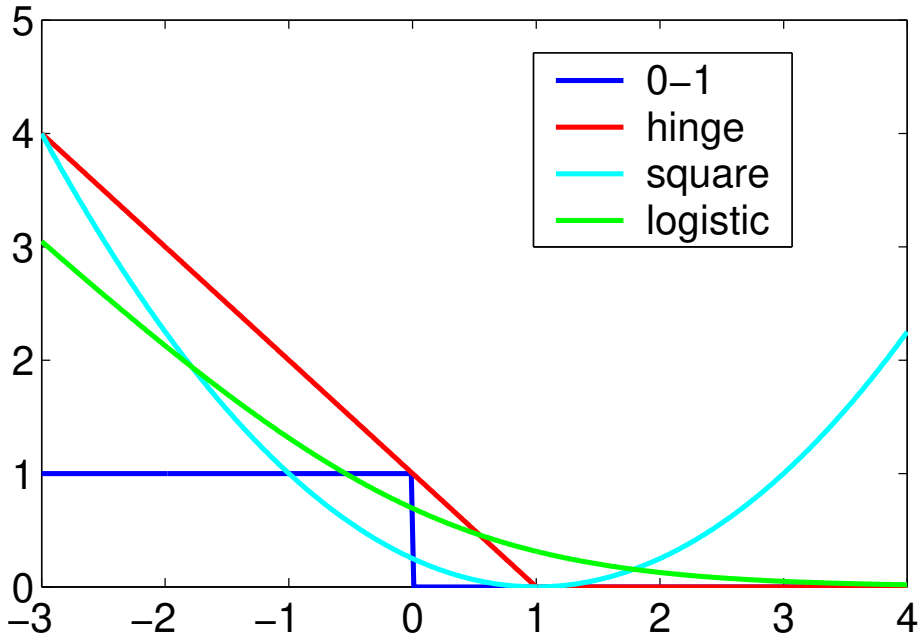
Usual losses

- **Regression**: $y \in \mathbb{R}$, prediction $\hat{y} = h(x, \theta)$
  - quadratic loss $\frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - h(x, \theta))^2$

Usual losses

- **Regression**: $y \in \mathbb{R}$, prediction $\hat{y} = h(x, \theta)$
  - quadratic loss $\frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - h(x, \theta))^2$

- **Classification** : $y \in \{-1, 1\}$, prediction $\hat{y} = \text{sign}(h(x, \theta))$
  - loss of the form $\ell(y\, h(x, \theta))$
  - "True" 0-1 loss: $\ell(y\, h(x, \theta)) = 1_{y\, h(x,\theta)<0}$
  - Usual convex losses:



Usual regularizers

- **Main goal**: avoid overfitting
- **(squared) Euclidean norm**: $\|\theta\|_2^2 = \sum_{j=1}^{d} |\theta_j|^2$
  - Numerically well-behaved if $h(x, \theta) = \theta^\top \Phi(x)$
  - Representer theorem and kernel methods : $\theta = \sum_{i=1}^{n} \alpha_i \Phi(x_i)$
  - See, e.g., Schölkopf and Smola (2001); Shawe-Taylor and Cristianini (2004)

Usual regularizers

- **Main goal**: avoid overfitting
- **(squared) Euclidean norm**: $\|\theta\|_2^2 = \sum_{j=1}^d |\theta_j|^2$
  - Numerically well-behaved if $h(x, \theta) = \theta^\top \Phi(x)$
  - Representer theorem and kernel methods : $\theta = \sum_{i=1}^n \alpha_i \Phi(x_i)$
  - See, e.g., Schölkopf and Smola (2001); Shawe-Taylor and Cristianini (2004)
- **Sparsity-inducing norms**
  - Main example: ℓ_1-norm $\|\theta\|_1 = \sum_{j=1}^d |\theta_j|$
  - Perform model selection as well as regularization
  - Non-smooth optimization and structured sparsity
  - See, e.g., Bach, Jenatton, Mairal, and Obozinski (2012a,b)

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**: find $\hat{\theta}$ solution of

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

data fitting term + regularizer

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 $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

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

data fitting term + regularizer

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 $\theta \in \mathbb{R}^d$
- **(regularized) empirical risk minimization**: find $\hat{\theta}$ solution of

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

data fitting term + regularizer

- Optimization: optimization of regularized risk training cost

Parametric supervised machine learning

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

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

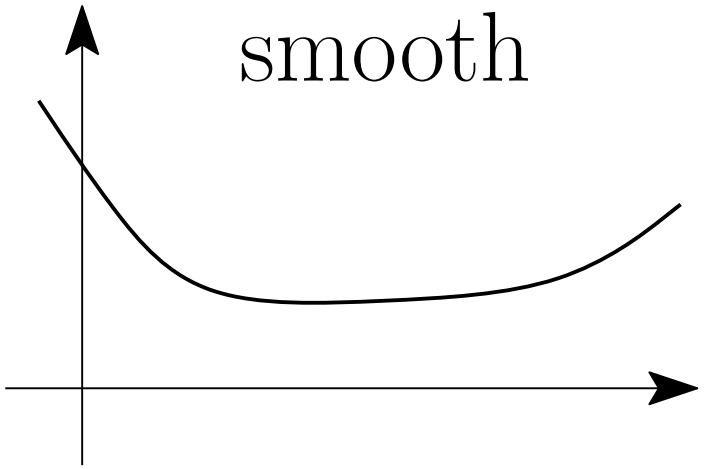
data fitting term + regularizer

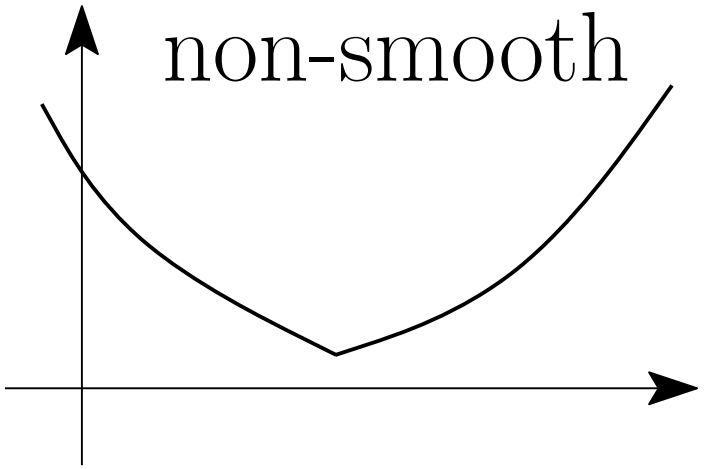
- Optimization: optimization of regularized risk — training cost
- Statistics: guarantees on $\mathbb{E}_{p(x,y)} \ell(y, h(x, \theta))$ — testing cost

Smoothness and (strong) convexity

- A function $g : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if and only if it is twice differentiable and

$$\forall \theta \in \mathbb{R}^d, \ \left|\text{eigenvalues}\left[g''(\theta)\right]\right| \leqslant L$$





Smoothness and (strong) convexity

- A function $g : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if and only if it is twice differentiable and

$$\forall \theta \in \mathbb{R}^d, \ \big|\text{eigenvalues}\big[g''(\theta)\big]\big| \leqslant L$$

- **Machine learning**
  - with $g(\theta) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Smooth prediction function $\theta \mapsto h(x_i, \theta)$ + smooth loss
  - *(see next slide)*

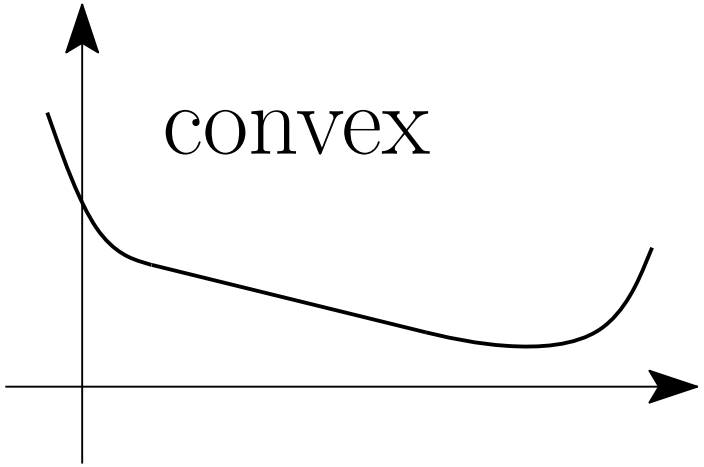
Board

- Function $g(\theta) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, \theta^\top \Phi(x_i))$
- Gradient $g'(\theta) = \frac{1}{n} \sum_{i=1}^n \ell'(y_i, \theta^\top \Phi(x_i)) \Phi(x_i)$
- Hessian $g''(\theta) = \frac{1}{n} \sum_{i=1}^n \ell''(y_i, \theta^\top \Phi(x_i)) \Phi(x_i) \Phi(x_i)^\top$
  - Smooth loss $\Rightarrow \ell''(y_i, \theta^\top \Phi(x_i))$ bounded

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is convex if and only if

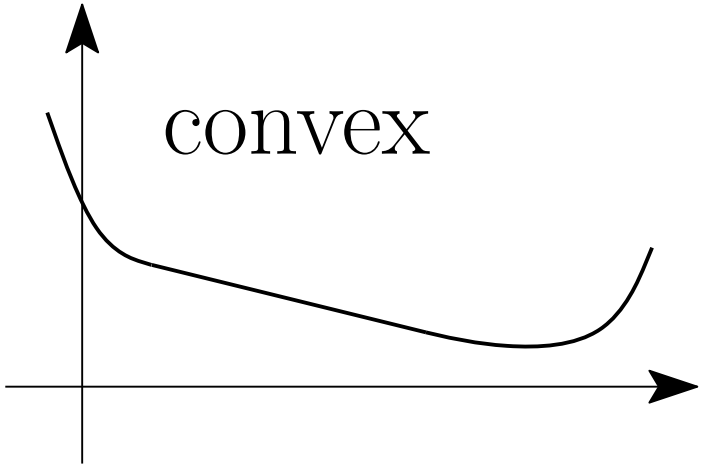
$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant 0$$

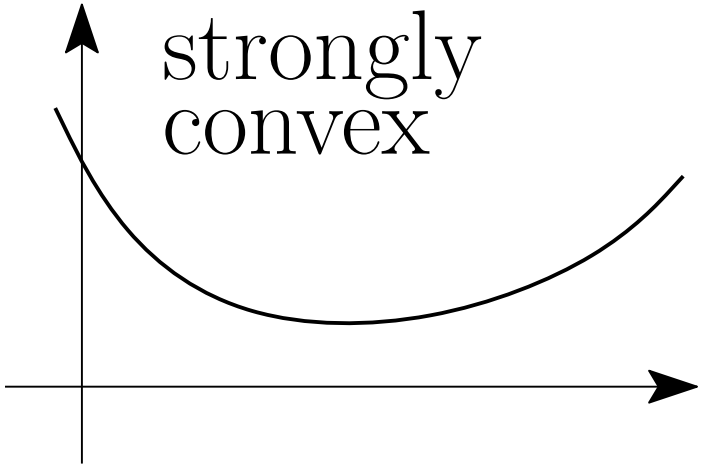


Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$



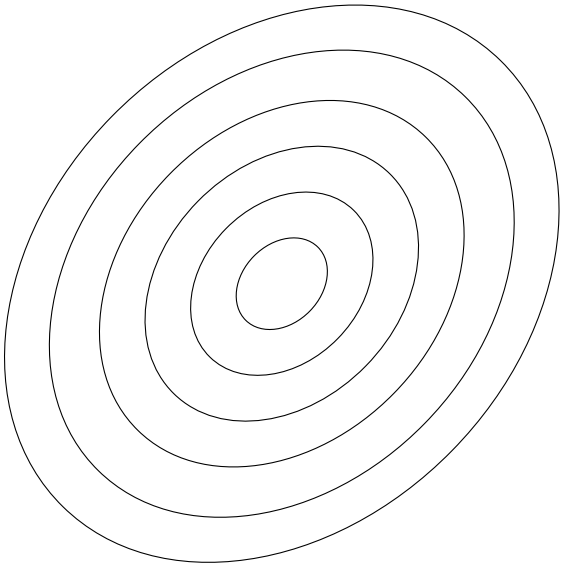


Smoothness and (strong) convexity

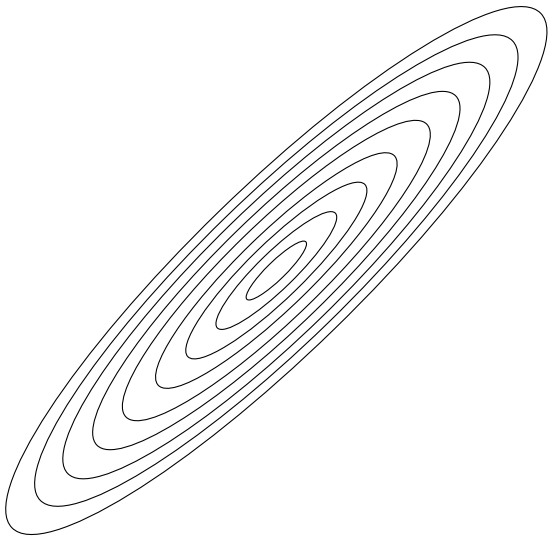
- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

  - Condition number $\kappa = L/\mu \geqslant 1$



(small $\kappa = L/\mu$)



(large $\kappa = L/\mu$)

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Convexity in machine learning**
  - With $g(\theta) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Convexity in machine learning**
  - With $g(\theta) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

- **Relevance of convex optimization**
  - Easier design and analysis of algorithms
  - Global minimum vs. local minimum vs. stationary points
  - Gradient-based algorithms only need convexity for their analysis

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Strong convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Strongly convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Strong convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Strongly convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$
  - Invertible covariance matrix $\frac{1}{n}\sum_{i=1}^n \Phi(x_i)\Phi(x_i)^\top \Rightarrow n \geqslant d$ *(slide)*
  - Even when $\mu > 0$, μ may be arbitrarily small!

Board

- Function $g(\theta) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, \theta^\top \Phi(x_i))$
- Gradient $g'(\theta) = \frac{1}{n} \sum_{i=1}^n \ell'(y_i, \theta^\top \Phi(x_i)) \Phi(x_i)$
- Hessian $g''(\theta) = \frac{1}{n} \sum_{i=1}^n \ell''(y_i, \theta^\top \Phi(x_i)) \Phi(x_i) \Phi(x_i)^\top$
  - Smooth loss $\Rightarrow \ell''(y_i, \theta^\top \Phi(x_i))$ bounded
- Square loss $\Rightarrow \ell''(y_i, \theta^\top \Phi(x_i)) = 1$
  - Hessian proportional to $\frac{1}{n} \sum_{i=1}^n \Phi(x_i) \Phi(x_i)^\top$

Smoothness and (strong) convexity

- A twice differentiable function $g : \mathbb{R}^d \to \mathbb{R}$ is μ-strongly convex if and only if

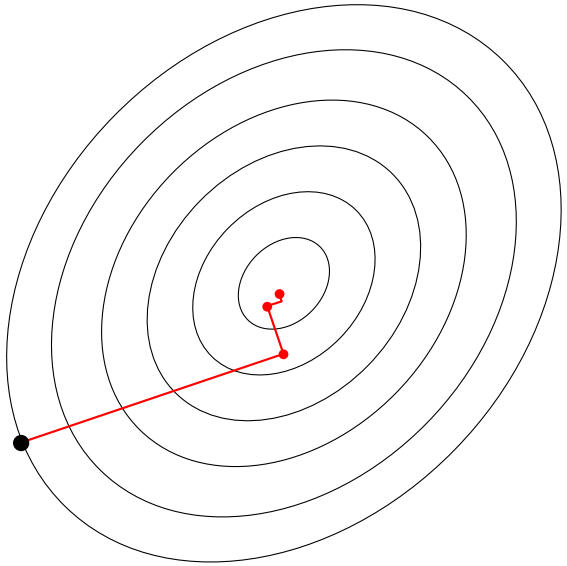
$$\forall \theta \in \mathbb{R}^d, \text{ eigenvalues}\big[g''(\theta)\big] \geqslant \mu$$

- **Strong convexity in machine learning**
  - With $g(\theta) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, h(x_i, \theta))$
  - Strongly convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$
  - Invertible covariance matrix $\frac{1}{n}\sum_{i=1}^n \Phi(x_i)\Phi(x_i)^\top \Rightarrow n \geqslant d$ *(slide)*
  - Even when $\mu > 0$, μ may be arbitrarily small!

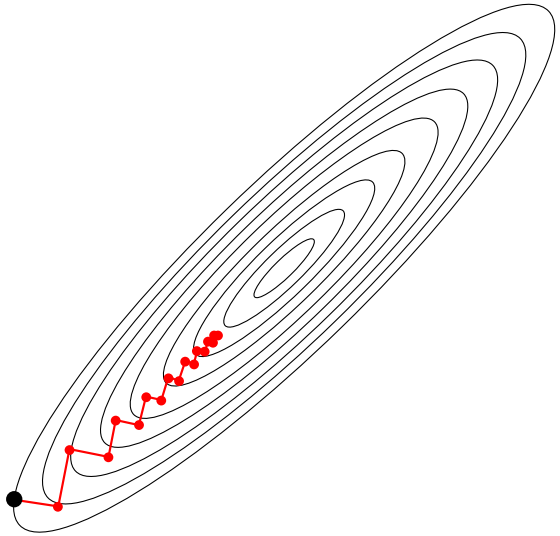
- **Adding regularization by $\frac{\mu}{2}\|\theta\|^2$**
  - creates additional bias unless μ is small, but reduces variance
  - Typically $\sqrt{n} \leqslant \kappa = L/\mu \leqslant n$

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$ *(line search)*



(small $\kappa = L/\mu$)

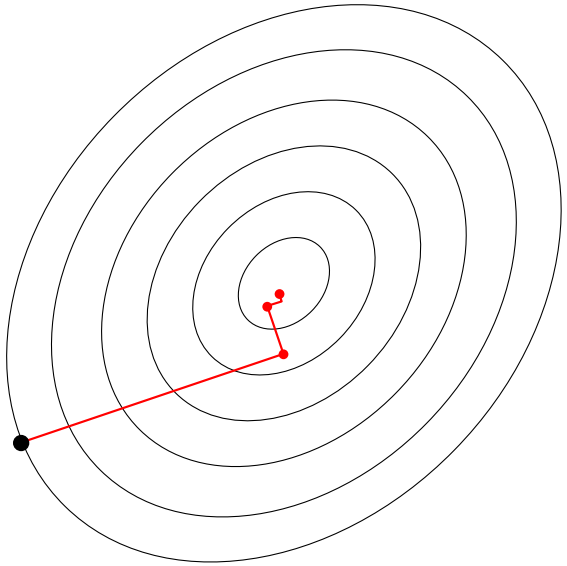


(large $\kappa = L/\mu$)

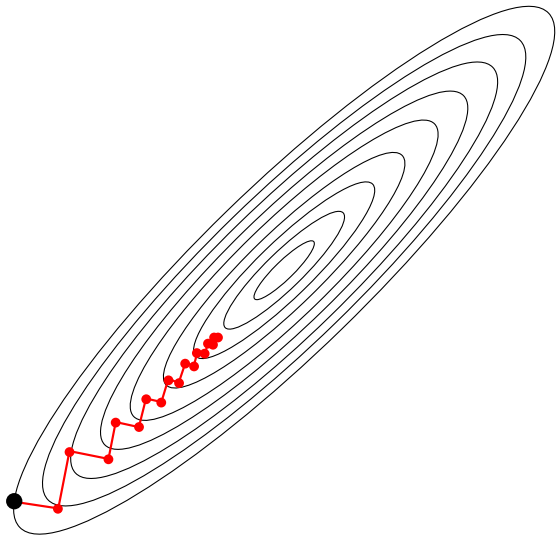
Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$ *(line search)*

$$g(\theta_t) - g(\theta_*) \leqslant O(1/t)$$
$$g(\theta_t) - g(\theta_*) \leqslant O((1-\mu/L)^t) = O(e^{-t(\mu/L)}) \text{ if } \mu\text{-strongly convex}$$



(small $\kappa = L/\mu$)



(large $\kappa = L/\mu$)

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ $O(\kappa \log \frac{1}{\varepsilon})$ iterations
- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ $O(\log \log \frac{1}{\varepsilon})$ iterations

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$
- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ complexity $= O((nd^2 + d^3) \cdot \log \log \frac{1}{\varepsilon})$

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$
- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ complexity $= O((nd^2 + d^3) \cdot \log \log \frac{1}{\varepsilon})$
- **Key insights for machine learning (Bottou and Bousquet, 2008)**
  1. No need to optimize below statistical error
  2. Cost functions are averages
  3. Testing error is more important than training error

Iterative methods for minimizing smooth functions

- **Assumption**: g convex and L-smooth on $\mathbb{R}^d$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma_t \, g'(\theta_{t-1})$
  - $O(1/t)$ convergence rate for convex functions
  - $O(e^{-t/\kappa})$ *linear* if strongly-convex $\Leftrightarrow$ complexity $= O(nd \cdot \kappa \log \frac{1}{\varepsilon})$
- **Newton method**: $\theta_t = \theta_{t-1} - g''(\theta_{t-1})^{-1} g'(\theta_{t-1})$
  - $O\big(e^{-\rho 2^t}\big)$ *quadratic* rate $\Leftrightarrow$ complexity $= O((nd^2 + d^3) \cdot \log \log \frac{1}{\varepsilon})$
- **Key insights for machine learning (Bottou and Bousquet, 2008)**
  1. No need to optimize below statistical error
  2. Cost functions are averages
  3. Testing error is more important than training error

Outline

1. **Introduction/motivation: Supervised machine learning**
   - Machine learning $\approx$ optimization of finite sums
   - Batch optimization methods

2. **Fast stochastic gradient methods for convex problems**
   - Variance reduction: for *training* error
   - Single pass SGD: for *testing* error

3. **Beyond convex problems**
   - Generic algorithms with generic "guarantees"
   - Global convergence for over-parameterized neural networks

Parametric supervised machine learning

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

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

data fitting term + regularizer

- Optimization: optimization of regularized risk — training cost
- Statistics: guarantees on $\mathbb{E}_{p(x,y)} \ell(y, h(x, \theta))$ — testing cost

Stochastic gradient descent (SGD) for finite sums

$$\min_{\theta \in \mathbb{R}^d} g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

- **Iteration**: $\theta_t = \theta_{t-1} - \gamma_t f'_{i(t)}(\theta_{t-1})$
  - Sampling with replacement: $i(t)$ random element of $\{1, \dots, n\}$
  - Polyak-Ruppert averaging: $\bar{\theta}_t = \frac{1}{t+1} \sum_{u=0}^{t} \theta_u$

Stochastic gradient descent (SGD) for finite sums

$$\min_{\theta \in \mathbb{R}^d} g(\theta) = \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

- **Iteration**: $\theta_t = \theta_{t-1} - \gamma_t f'_{i(t)}(\theta_{t-1})$
  - Sampling with replacement: $i(t)$ random element of $\{1, \dots, n\}$
  - Polyak-Ruppert averaging: $\bar{\theta}_t = \frac{1}{t+1} \sum_{u=0}^{t} \theta_u$

- **Convergence rate** if each f_i is convex L-smooth and g μ-strongly-convex:

$$\mathbb{E} g(\bar{\theta}_t) - g(\theta_*) \leqslant \begin{cases} O(1/\sqrt{t}) & \text{if } \gamma_t = 1/(L\sqrt{t}) \\ O(L/(\mu t)) = O(\kappa/t) & \text{if } \gamma_t = 1/(\mu t) \end{cases}$$

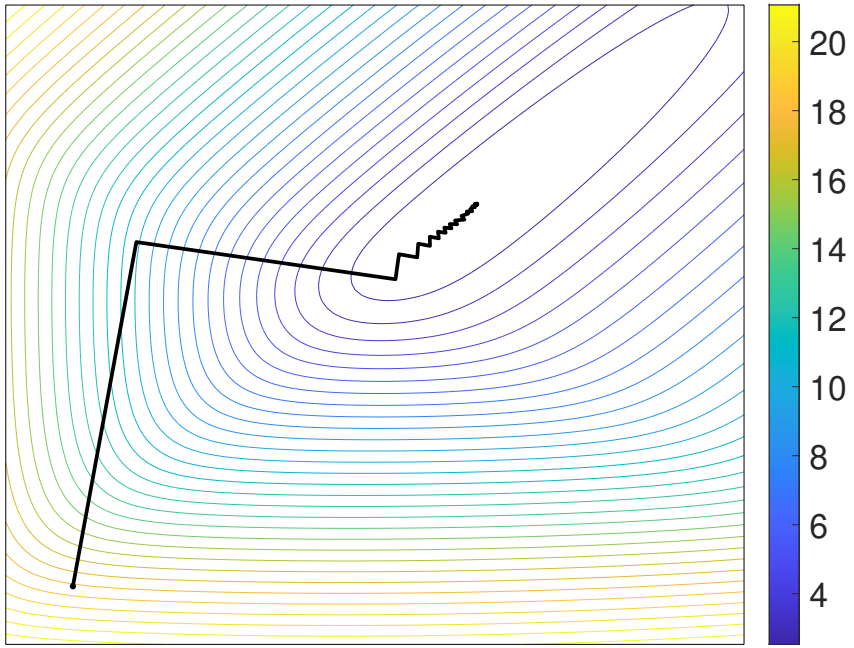
  - No adaptivity to strong-convexity in general
  - Running-time complexity: $O(d \cdot \kappa/\varepsilon)$

Deterministic and stochastic methods

- Minimize $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i, \theta)\big) + \lambda\Omega(\theta)$

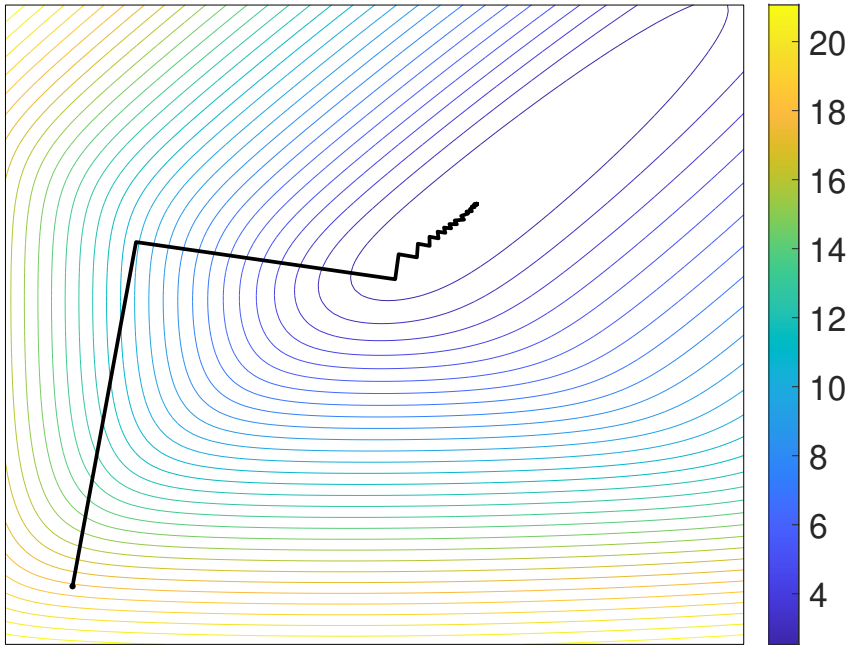
Deterministic and stochastic methods

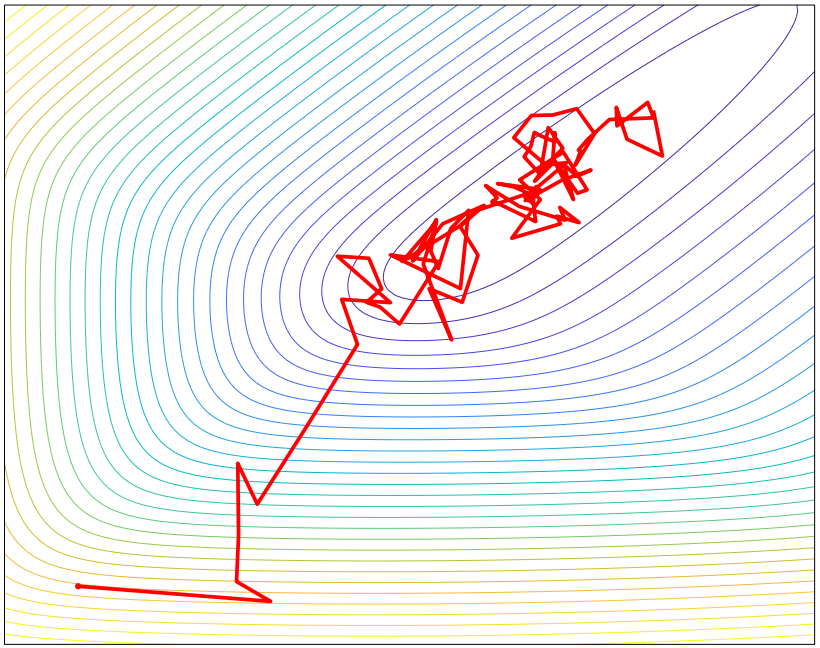
- Minimize $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i, \theta)\big) + \lambda\Omega(\theta)$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma\nabla g(\theta_{t-1}) = \theta_{t-1} - \frac{\gamma}{n}\sum_{i=1}^{n}\nabla f_i(\theta_{t-1})$ (Cauchy, 1847)



Deterministic and stochastic methods

- Minimize $g(\theta) = \frac{1}{n}\sum_{i=1}^{n} f_i(\theta)$ with $f_i(\theta) = \ell\big(y_i, h(x_i, \theta)\big) + \lambda\Omega(\theta)$
- **Gradient descent**: $\theta_t = \theta_{t-1} - \gamma\nabla g(\theta_{t-1}) = \theta_{t-1} - \frac{\gamma}{n}\sum_{i=1}^{n}\nabla f_i(\theta_{t-1})$ (Cauchy, 1847)
- **Stochastic gradient descent**: $\theta_t = \theta_{t-1} - \gamma\nabla f_{i(t)}(\theta_{t-1})$ (Robbins and Monro, 1951)

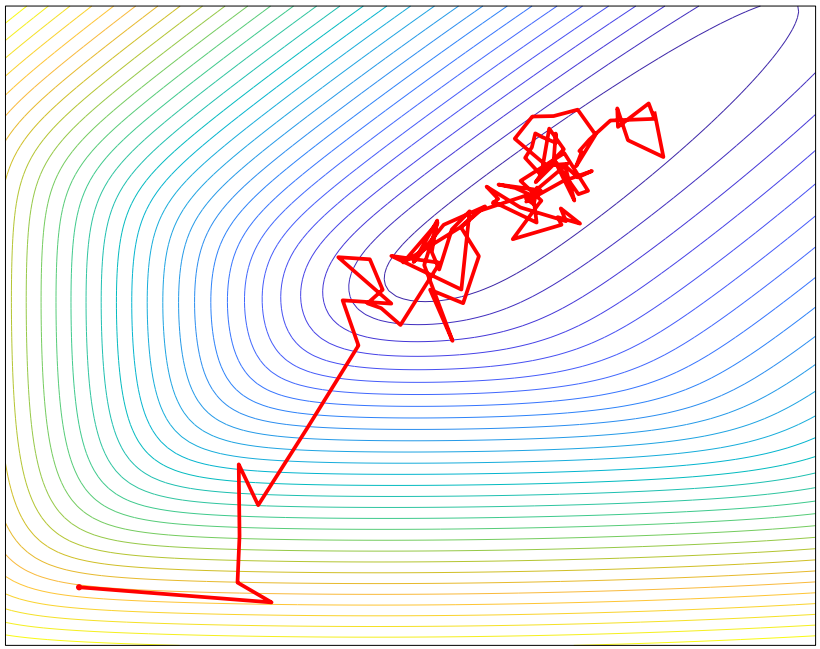




Stochastic gradient with exponential convergence

- **Variance reduction**
  - SAG (Le Roux, Schmidt, and Bach, 2012)
  - SVRG (Johnson and Zhang, 2013; Zhang et al., 2013)
  - SAGA (Defazio, Bach, and Lacoste-Julien, 2014)

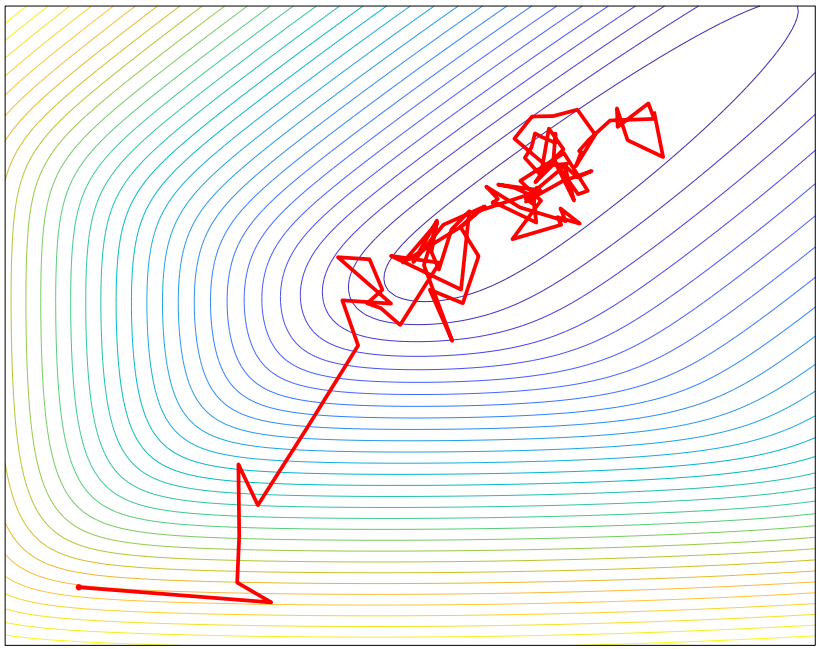
$$\theta_t = \theta_{t-1} - \gamma \left[\nabla f_{i(t)}(\theta_{t-1}) \right]$$

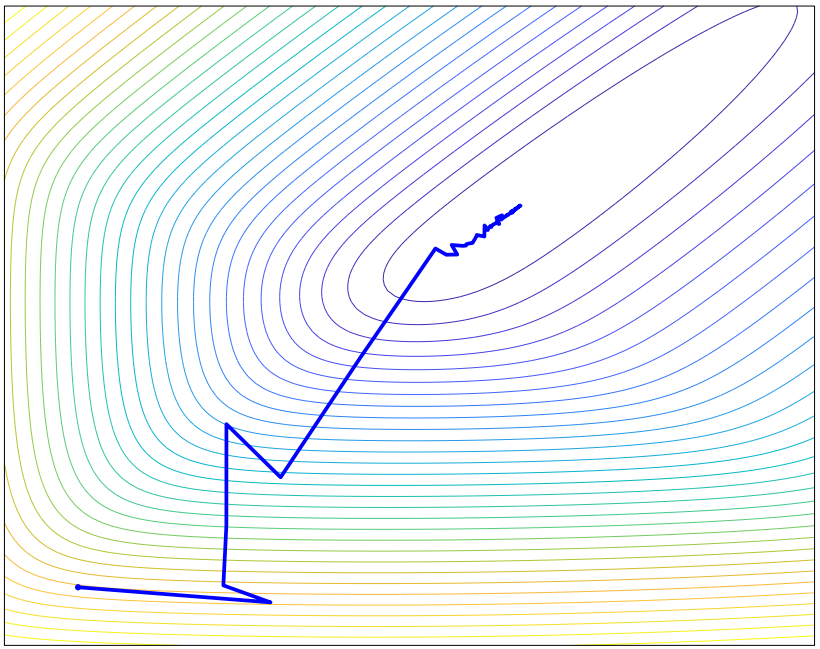


Stochastic gradient with exponential convergence

- **Variance reduction**
  - SAG (Le Roux, Schmidt, and Bach, 2012)
  - SVRG (Johnson and Zhang, 2013; Zhang et al., 2013)
  - SAGA (Defazio, Bach, and Lacoste-Julien, 2014)

$$\theta_t = \theta_{t-1} - \gamma \left[\nabla f_{i(t)}(\theta_{t-1}) + \frac{1}{n} \sum_{i=1}^{n} y_i^{t-1} - y_{i(t)}^{t-1} \right]$$





Stochastic gradient with exponential convergence

- **Variance reduction**
  - SAG (Le Roux, Schmidt, and Bach, 2012)
  - SVRG (Johnson and Zhang, 2013; Zhang et al., 2013)
  - SAGA (Defazio, Bach, and Lacoste-Julien, 2014)

- **Number of individual gradient computations to reach error** ε (convex objectives with condition number κ)

Gradient descent	$n\kappa \quad \times \log \frac{1}{\varepsilon}$
Stochastic gradient descent	$\kappa \quad \times \quad \frac{1}{\varepsilon}$
Variance reduction	$(n+\kappa) \quad \times \log \frac{1}{\varepsilon}$

Stochastic gradient with exponential convergence

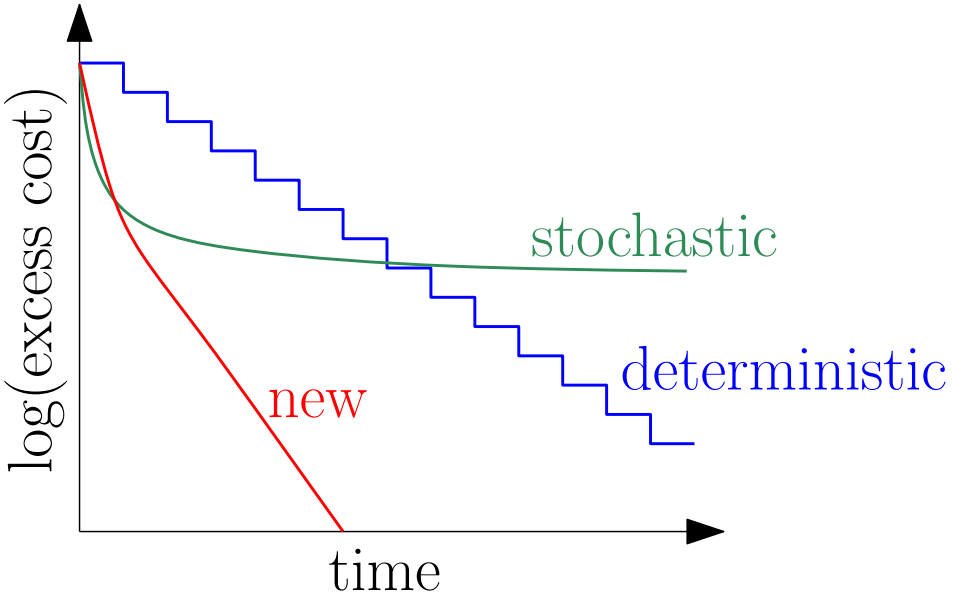
- **Variance reduction**
  - SAG (Le Roux, Schmidt, and Bach, 2012)
  - SVRG (Johnson and Zhang, 2013; Zhang et al., 2013)
  - SAGA (Defazio, Bach, and Lacoste-Julien, 2014)

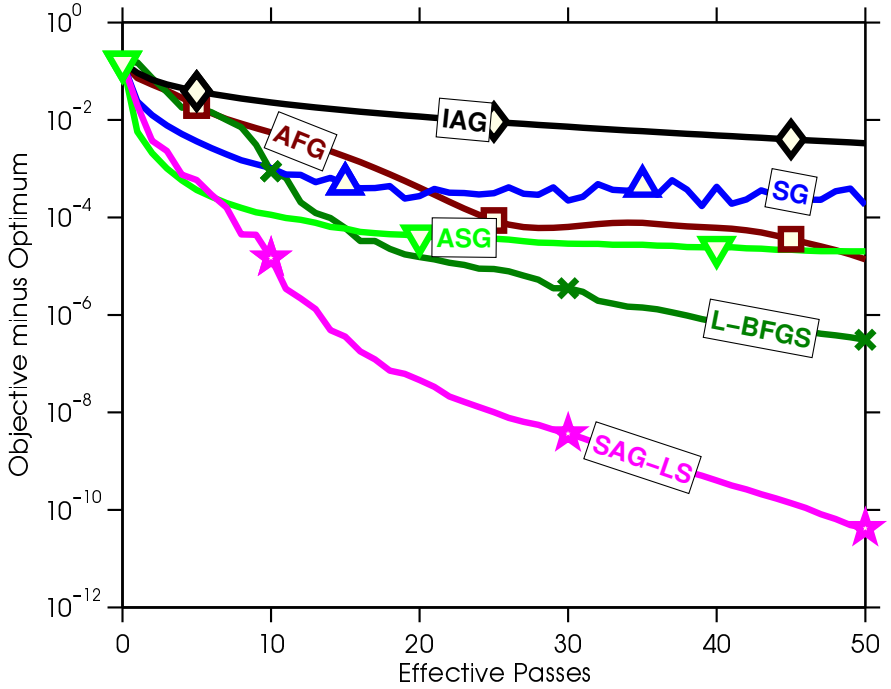
- **Number of individual gradient computations to reach error** ε (convex objectives with condition number κ)

Gradient descent	$n\kappa \quad \times \log \frac{1}{\varepsilon}$
Stochastic gradient descent	$\kappa \quad \times \quad \frac{1}{\varepsilon}$
Variance reduction	$(n+\kappa) \quad \times \log \frac{1}{\varepsilon}$

- **Empirical behavior close to complexity bounds**

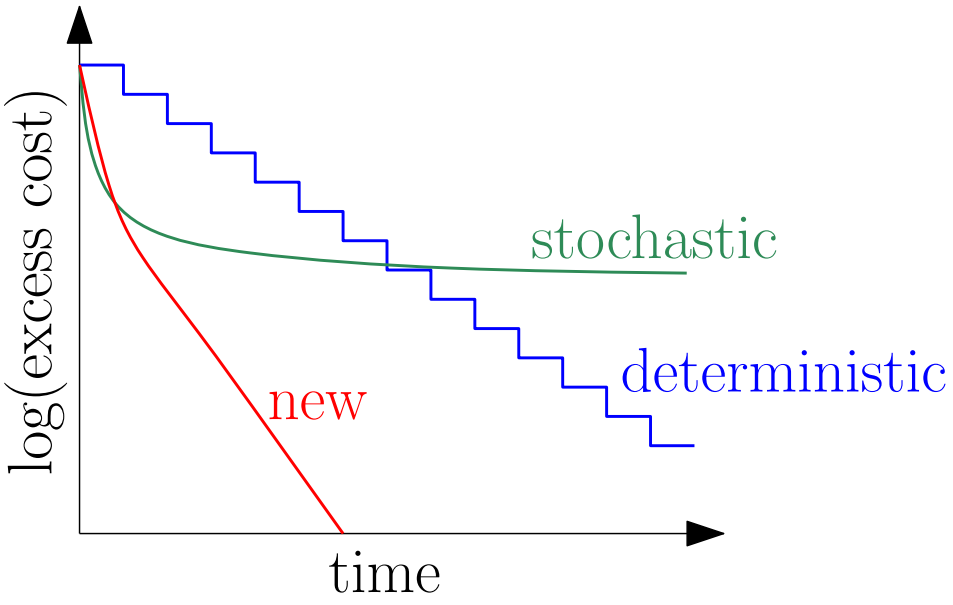
Exponentially convergent SGD for finite sums
From theory to practice and vice-versa

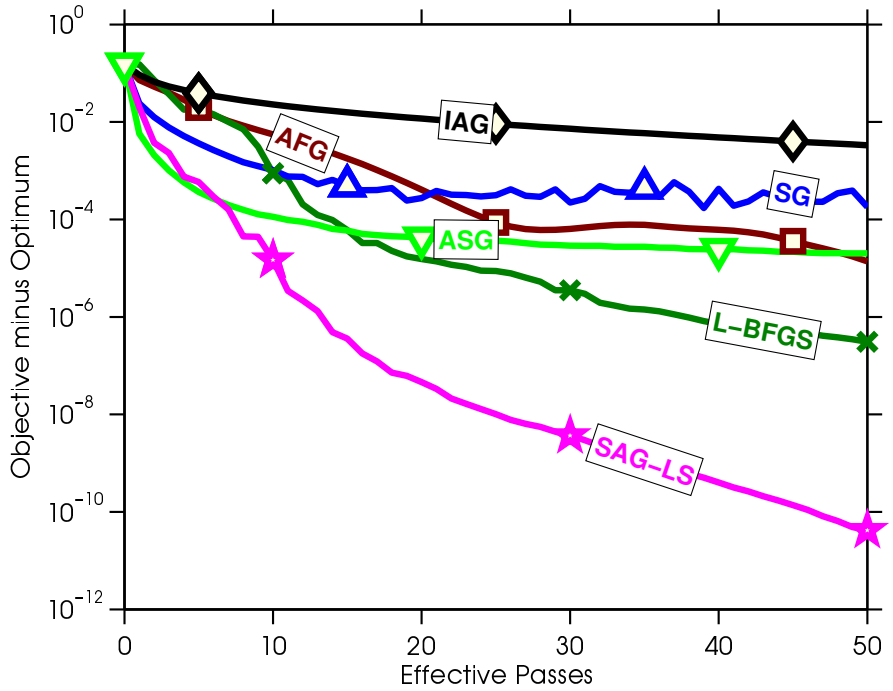




- **Empirical performance "matches" theoretical guarantees**

Exponentially convergent SGD for finite sums
From theory to practice and vice-versa

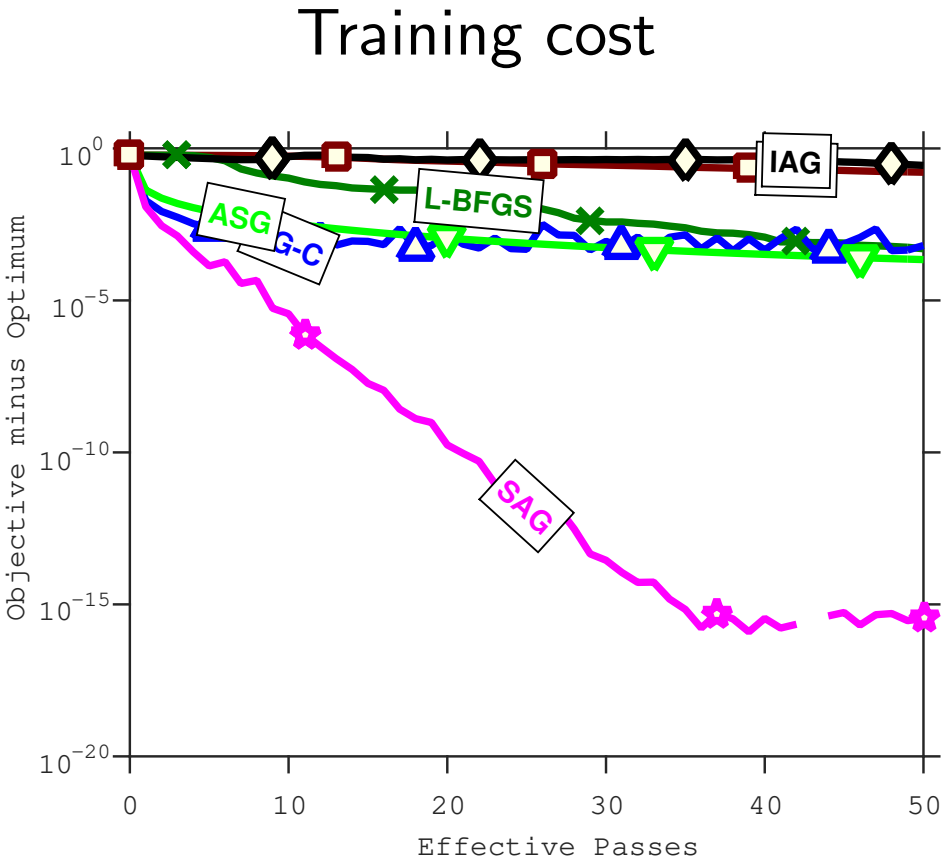




- **Empirical performance "matches" theoretical guarantees**
- **Theoretical analysis suggests practical improvements**
  - Non-uniform sampling, acceleration
  - Matching upper and lower bounds

From training to testing errors

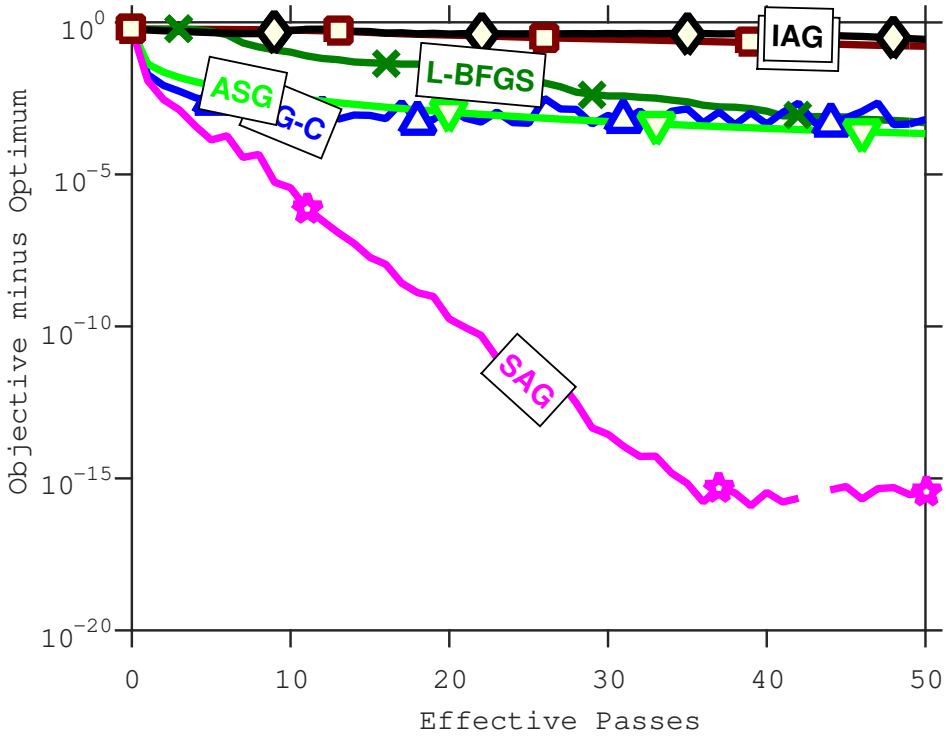
- `rcv1` dataset ($n = 697\,641$, $d = 47\,236$)
  - NB: IAG, SG-C, ASG with optimal step-sizes in hindsight



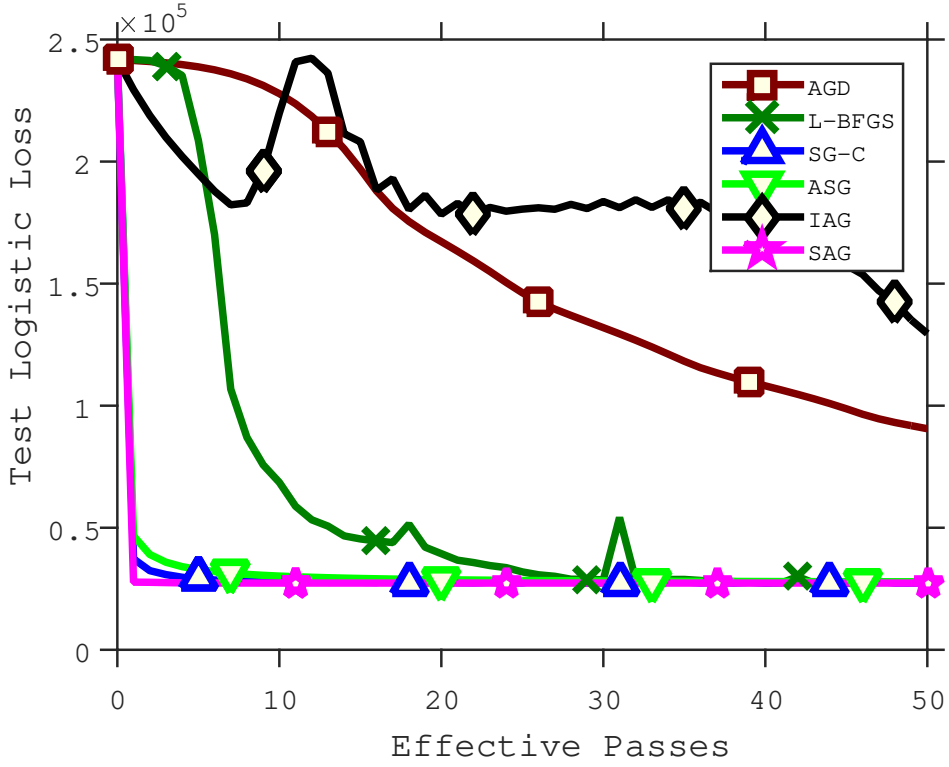
From training to testing errors

- `rcv1` dataset ($n = 697\,641$, $d = 47\,236$)
  - NB: IAG, SG-C, ASG with optimal step-sizes in hindsight

Training cost



Testing cost



SGD minimizes the testing cost!

- **Goal**: minimize $f(\theta) = \mathbb{E}_{p(x,y)}\ell(y, h(x, \theta))$
  - Given n independent samples (x_i, y_i), $i = 1, \ldots, n$ from $p(x, y)$
  - Given a single pass of stochastic gradient descent
  - Bounds on the excess testing cost $\mathbb{E}f(\bar{\theta}_n) - \inf_{\theta \in \mathbb{R}^d} f(\theta)$

SGD minimizes the testing cost!

- **Goal**: minimize $f(\theta) = \mathbb{E}_{p(x,y)}\ell(y, h(x, \theta))$
  - Given n independent samples (x_i, y_i), $i = 1, \ldots, n$ from $p(x, y)$
  - Given a single pass of stochastic gradient descent
  - Bounds on the excess testing cost $\mathbb{E}f(\bar{\theta}_n) - \inf_{\theta \in \mathbb{R}^d} f(\theta)$

- **Optimal convergence rates**: $O(1/\sqrt{n})$ and $O(1/(n\mu))$
  - Optimal for non-smooth losses (Nemirovski and Yudin, 1983)
  - Attained by averaged SGD with decaying step-sizes

SGD minimizes the testing cost!

- **Goal**: minimize $f(\theta) = \mathbb{E}_{p(x,y)}\ell(y, h(x, \theta))$
  - Given n independent samples (x_i, y_i), $i = 1, \ldots, n$ from $p(x, y)$
  - Given a single pass of stochastic gradient descent
  - Bounds on the excess testing cost $\mathbb{E}f(\bar{\theta}_n) - \inf_{\theta \in \mathbb{R}^d} f(\theta)$

- **Optimal convergence rates**: $O(1/\sqrt{n})$ and $O(1/(n\mu))$
  - Optimal for non-smooth losses (Nemirovski and Yudin, 1983)
  - Attained by averaged SGD with decaying step-sizes

- **Constant-step-size SGD**
  - Convergence up to the noise level (Solodov, 1998)
  - Full convergence and robustness to ill-conditioning (Bach and Moulines, 2013)

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations
- **Extensions and future work**
  - Matching lower bounds (Woodworth and Srebro, 2016; Lan, 2015)
  - Sampling without replacement (Gurbuzbalaban et al., 2015)

Perspectives

- **Linearly-convergent stochastic gradient methods**
  - Provable and precise rates
  - Improves on two known lower-bounds (by using structure)
  - Several extensions / interpretations / accelerations

- **Extensions and future work**
  - Matching lower bounds (Woodworth and Srebro, 2016; Lan, 2015)
  - Sampling without replacement (Gurbuzbalaban et al., 2015)
  - Parallelization (Leblond, Pedregosa, and Lacoste-Julien, 2016; Hendrikx, Bach, and Massoulié, 2019)
  - Non-convex problems (Reddi et al., 2016)

Outline

1. **Introduction/motivation: Supervised machine learning**
   - Machine learning $\approx$ optimization of finite sums
   - Batch optimization methods

2. **Fast stochastic gradient methods for convex problems**
   - Variance reduction: for *training* error
   - Single pass SGD: for *testing* error

2. **Beyond convex problems**
   - Generic algorithms with generic "guarantees"
   - Global convergence for over-parameterized neural networks

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**:

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

data fitting term + regularizer

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

Convex optimization problems

- **Convexity in machine learning**
  - Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$

Convex optimization problems

- **Convexity in machine learning**
  - Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$
- **(approximately) matching theory and practice**
  - Fruitful discussions between theoreticians and practitioners
  - Quantitative theoretical analysis suggests practical improvements

Convex optimization problems

- **Convexity in machine learning**
  - Convex loss and linear predictions $h(x, \theta) = \theta^\top \Phi(x)$
- **(approximately) matching theory and practice**
  - Fruitful discussions between theoreticians and practitioners
  - Quantitative theoretical analysis suggests practical improvements
- **Golden years of convexity in machine learning** (1995 to 2020+)
  - Support vector machines and kernel methods
  - Inference in graphical models
  - Sparsity / low-rank models (statistics + optimization)
  - Convex relaxation of unsupervised learning problems
  - Optimal transport
  - Stochastic methods for large-scale learning and online learning

Convex optimization for machine learning
From theory to practice and vice-versa

- **Empirical performance "matches" theoretical guarantees**
- **Theoretical analysis suggests practical improvements**

Convex optimization for machine learning
From theory to practice and vice-versa

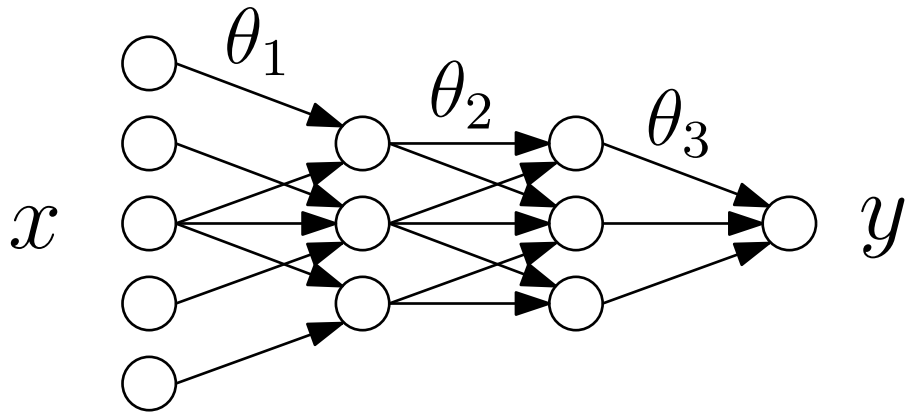
- **Empirical performance "matches" theoretical guarantees**
- **Theoretical analysis suggests practical improvements**
- **Many other well-understood areas**
  - Single pass SGD and generalization errors
  - From least-squares to convex losses
  - High-dimensional inference
  - Non-parametric regression
  - Randomized linear algebra
  - Bandit problems
  - etc...

Convex optimization for machine learning
From theory to practice and vice-versa

- **Empirical performance "matches" theoretical guarantees**
- **Theoretical analysis suggests practical improvements**
- **Many other well-understood areas**
  - Single pass SGD and generalization errors
  - From least-squares to convex losses
  - High-dimensional inference
  - Non-parametric regression
  - Randomized linear algebra
  - Bandit problems
  - etc...
- **What about deep learning?**

Theoretical analysis of deep learning

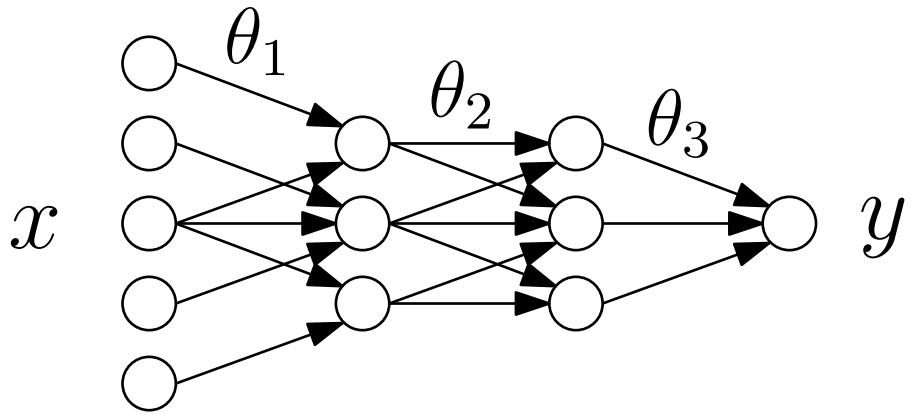
- **Multi-layer neural network** $h(x, \theta) = \theta_m^\top \sigma(\theta_{m-1}^\top \sigma(\cdots \theta_2^\top \sigma(\theta_1^\top x))$



  - NB: already a simplification

Theoretical analysis of deep learning

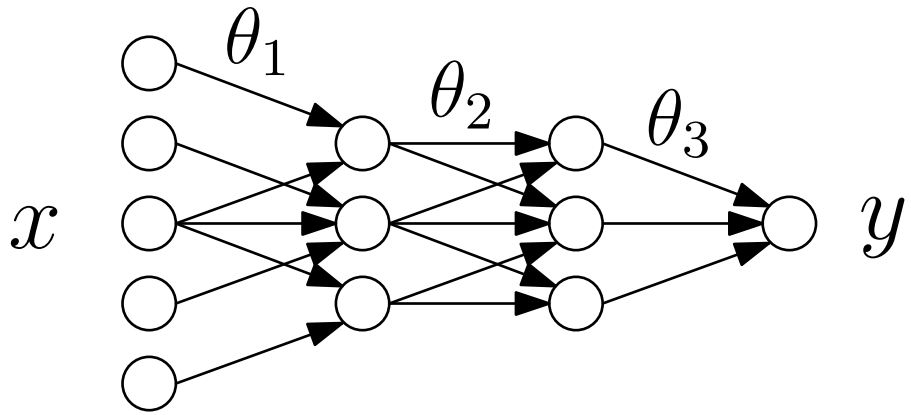
- **Multi-layer neural network** $h(x, \theta) = \theta_m^\top \sigma(\theta_{m-1}^\top \sigma(\cdots \theta_2^\top \sigma(\theta_1^\top x))$



- **Generalization guarantees**
  - See "MythBusters: A Deep Learning Edition" by Sasha Rakhlin
  - Bartlett et al. (2017); Golowich et al. (2018)

Theoretical analysis of deep learning

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

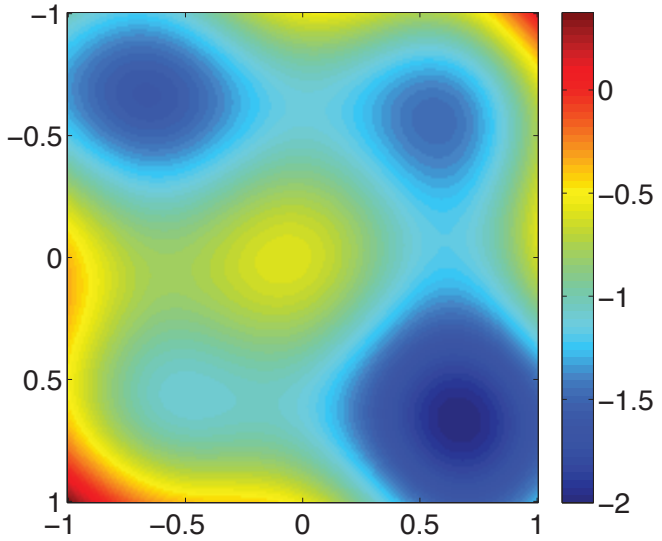


- **Generalization guarantees**
  - See "MythBusters: A Deep Learning Edition" by Sasha Rakhlin
  - Bartlett et al. (2017); Golowich et al. (2018)

- **Optimization**
  - Non-convex optimization problems

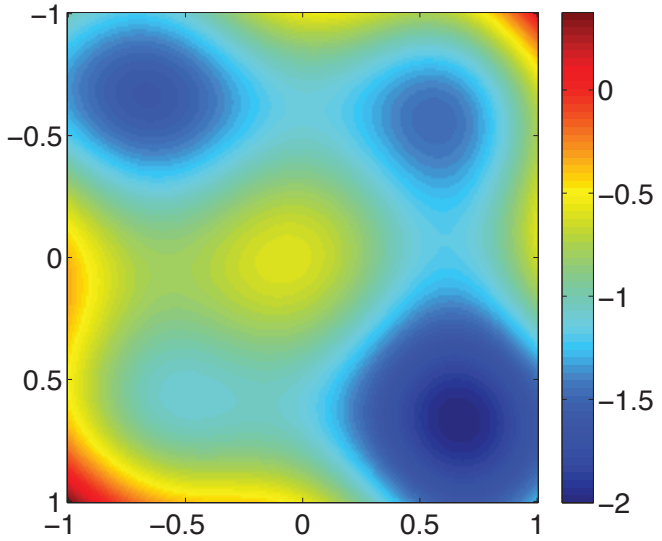
Optimization for multi-layer neural networks

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



Optimization for multi-layer neural networks

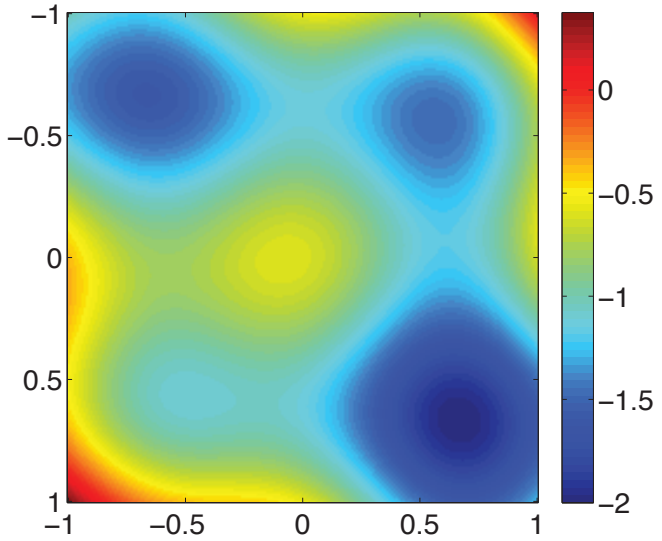
- **What can go wrong with non-convex optimization problems?**
  - Local minima
  - Stationary points
  - Plateaux
  - Bad initialization
  - etc...



- **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...

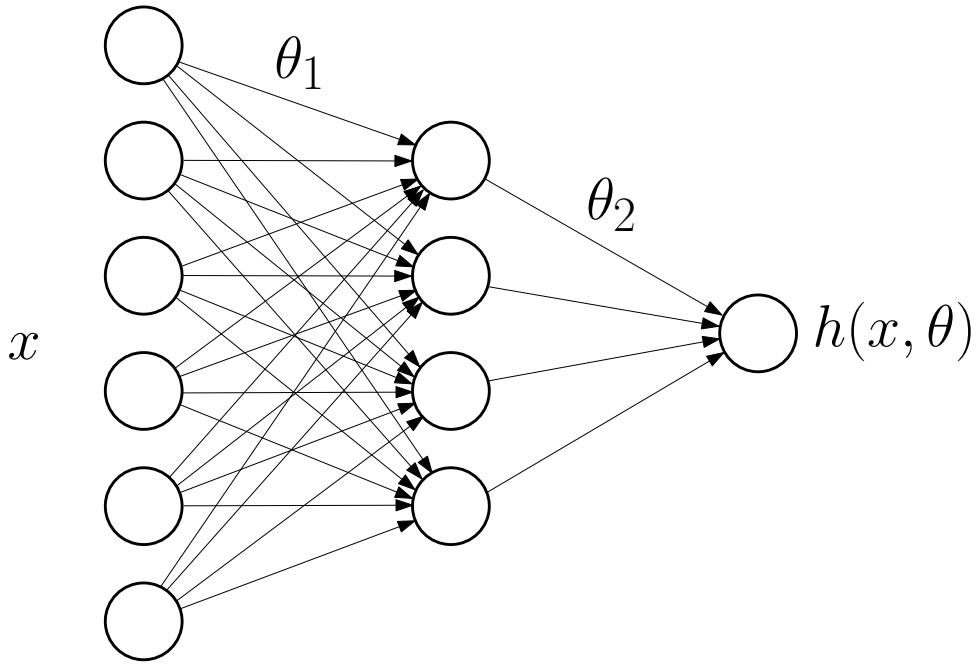


- **General global performance guarantees impossible to obtain**

Gradient descent for a single hidden layer

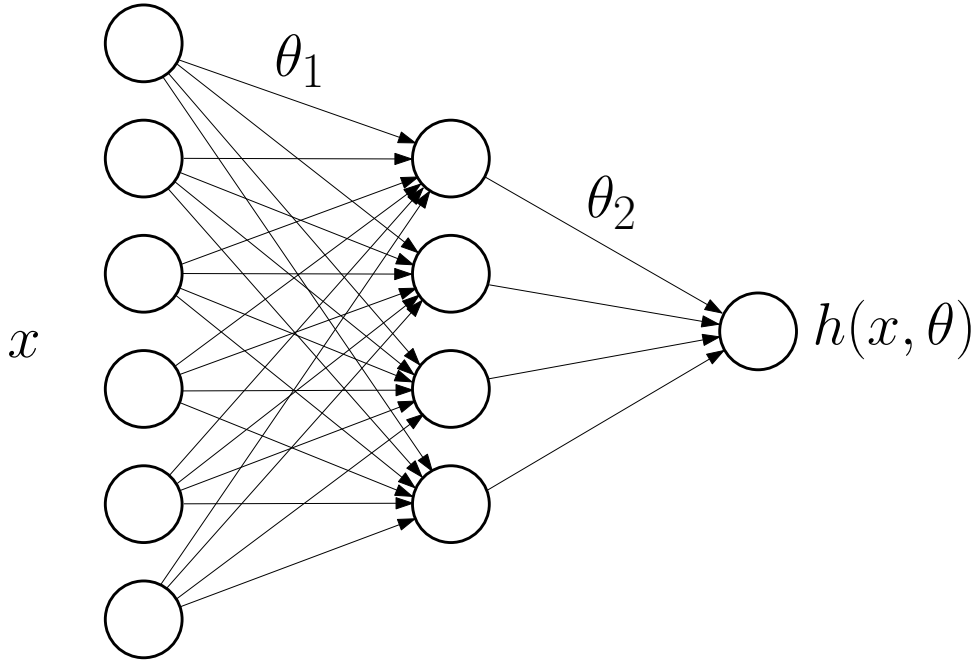
- **Predictor**: $h(x) = \theta_2^\top \sigma(\theta_1^\top x) = \sum_{i=1}^m \theta_2(i) \cdot \sigma\left[\theta_1(\cdot, i)^\top x\right]$

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



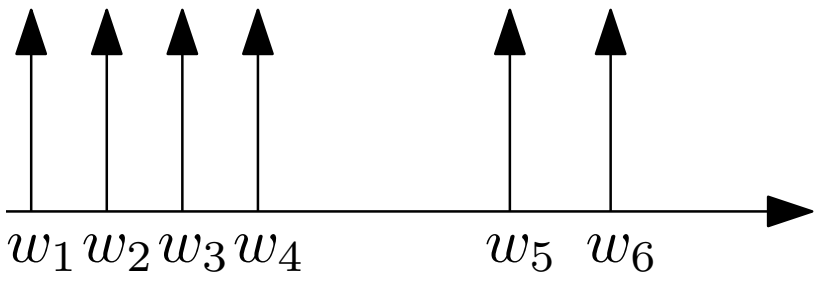
Gradient descent for a single hidden layer

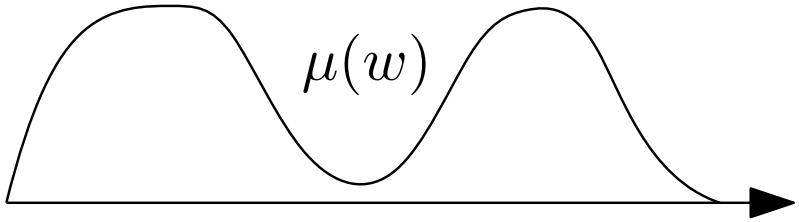
- **Predictor**: $h(x) = \theta_2^\top \sigma(\theta_1^\top x) = \sum_{i=1}^m \theta_2(i) \cdot \sigma\left[\theta_1(\cdot, i)^\top x\right]$
  - Family: $h = \frac{1}{m} \sum_{i=1}^m \Psi(w_i)$ with $\Psi(w_i)(x) = m\theta_2(i) \cdot \sigma\left[\theta_1(\cdot, i)^\top x\right]$
- **Goal**: minimize $R(h) = \mathbb{E}_{p(x,y)} \ell(y, h(x))$, with R convex



Gradient descent for a single hidden layer

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





Gradient descent for a single hidden layer

- **Predictor**: $h(x) = \theta_2^\top \sigma(\theta_1^\top x) = \sum_{i=1}^m \theta_2(i) \cdot \sigma\big[\theta_1(\cdot, i)^\top x\big]$
  - Family: $h = \dfrac{1}{m}\displaystyle\sum_{i=1}^m \Psi(w_i)$ with $\Psi(w_i)(x) = m\theta_2(i) \cdot \sigma\big[\theta_1(\cdot, i)^\top x\big]$
- **Goal**: minimize $R(h) = \mathbb{E}_{p(x,y)} \ell(y, h(x))$, with R convex
- **Main insight**
  - $h = \dfrac{1}{m}\displaystyle\sum_{i=1}^m \Psi(w_i) = \int_{\mathcal{W}} \Psi(w) d\mu(w)$ with $d\mu(w) = \dfrac{1}{m}\displaystyle\sum_{i=1}^m \delta_{w_i}$
  - 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"** $\mu = \frac{1}{m}\sum_{i=1}^{m} \delta_{w_i}$
  - Backpropagation = gradient descent on $(w_1, \ldots, w_m)$

- **Two questions**:
  - Algorithm limit when number of particles m gets large
  - Global convergence

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 $F_m(w_1, \ldots, w_m) = R\Big(\frac{1}{m}\sum_{i=1}^{m} \Psi(w_i)\Big)$

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 $F_m(w_1, \ldots, w_m) = R\Big(\frac{1}{m}\sum_{i=1}^{m} \Psi(w_i)\Big)$
  - Gradient flow $\dot{W} = -m\nabla F_m(W)$, with $W = (w_1, \ldots, w_m)$
  - 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 $F_m(w_1, \ldots, w_m) = R\Big(\frac{1}{m}\sum_{i=1}^{m} \Psi(w_i)\Big)$
  - Gradient flow $\dot{W} = -m\nabla F_m(W)$, with $W = (w_1, \ldots, w_m)$
  - 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)

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

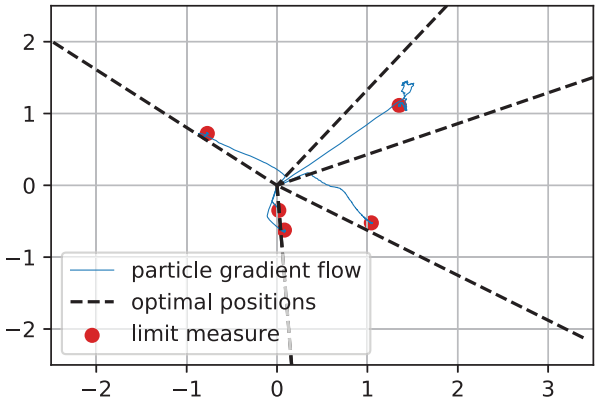
- **Two ingredients**: homogeneity and initialization

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

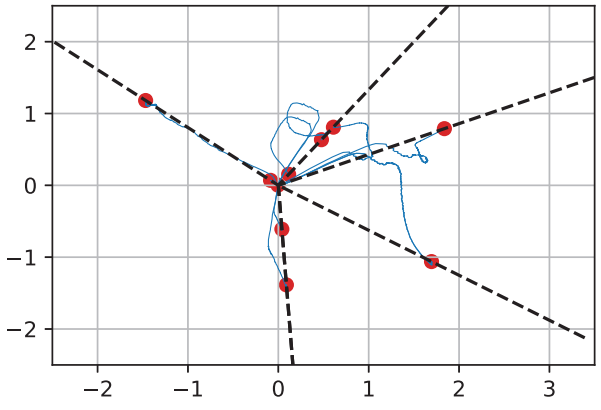
- **Two ingredients**: homogeneity and initialization
- **Homogeneity** (see, e.g., Haeffele and Vidal, 2017; Bach et al., 2008)
  - Full or partial, e.g., $\Psi(w_i)(x) = m\theta_2(i) \cdot \sigma\left[\theta_1(\cdot, i)^\top x\right]$
  - Applies to rectified linear units (but also to sigmoid activations)
- **Sufficiently spread initial measure**
  - Needs to cover the entire sphere of directions

Simple simulations with neural networks

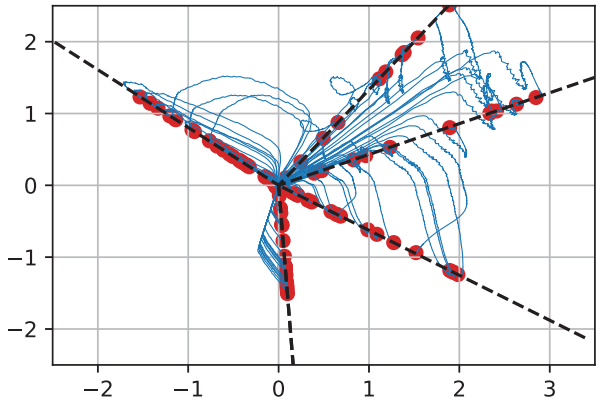
- ReLU units with $d = 2$ (optimal predictor has 5 neurons)



5 neurons



10 neurons



100 neurons

Conclusions
Optimization for machine learning

- **Well understood**
  - Convex case with a single machine
  - Matching lower and upper bounds for variants of SGD
  - Non-convex case: SGD for local risk minimization

Conclusions
Optimization for machine learning

- **Well understood**
  - Convex case with a single machine
  - Matching lower and upper bounds for variants of SGD
  - Non-convex case: SGD for local risk minimization

- **Not well understood**: many open problems
  - Step-size schedules and acceleration, conditioning
  - Dealing with non-convexity
    (global minima vs. local minima and stationary points)
  - Distributed learning: multiple cores, GPUs, and cloud
  - Beyond running time

References

F. Bach and E. Moulines. Non-strongly-convex smooth stochastic approximation with convergence rate $O(1/n)$. In *Adv. NIPS*, 2013.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012a.

F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Structured sparsity through convex optimization, 2012b.

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.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6240–6249, 2017.

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

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Adv. NIPS*, 2008.

M. A. Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Comptes rendus des séances de l'Académie des sciences*, 25(1):536–538, 1847.

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.

A. Defazio, F. Bach, and S. Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory*, pages 297–299, 2018.

M. Gurbuzbalaban, A. Ozdaglar, and P. Parrilo. On the convergence rate of incremental aggregated gradient algorithms. Technical Report 1506.02081, arXiv, 2015.

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.

Hadrien Hendrikx, Francis Bach, and Laurent Massoulié. Asynchronous accelerated proximal stochastic gradient for strongly convex distributed finite sums. Technical Report 1901.09865, arXiv, 2019.

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.

R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 2013.

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.

G. Lan. An optimal randomized incremental gradient method. Technical Report 1507.02000, arXiv,

2015.

N. Le Roux, M. Schmidt, and F. Bach. A stochastic gradient method with an exponential convergence rate for strongly-convex optimization with finite training sets. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

R. Leblond, F. Pedregosa, and S. Lacoste-Julien. Asaga: Asynchronous parallel Saga. Technical Report 1606.04809, arXiv, 2016.

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.

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.

A. S. Nemirovski and D. B. Yudin. *Problem complexity and method efficiency in optimization.* Wiley & Sons, 1983.

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

S. J. Reddi, A. Hefny, S. Sra, B. Póczós, and A. Smola. Stochastic variance reduction for nonconvex optimization. Technical Report 1603.06160, arXiv, 2016.

H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Statistics*, 22:400–407, 1951.

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.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

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

M. V. Solodov. Incremental gradient algorithms with stepsizes bounded away from zero. *Computational Optimization and Applications*, 11(1):23–35, 1998.

Blake E. Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in neural information processing systems*, pages 3639–3647, 2016.

L. Zhang, M. Mahdavi, and R. Jin. Linear convergence with condition number independent access of full gradients. In *Advances in Neural Information Processing Systems*, 2013.