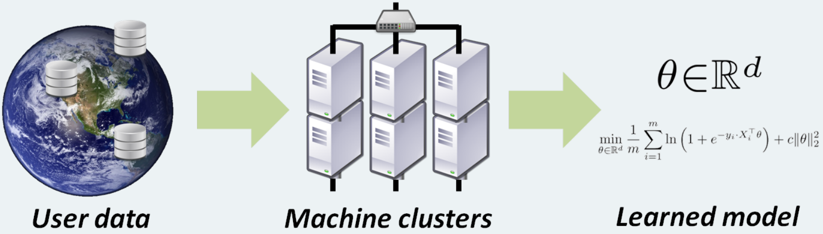
# Optimal convergence rates for distributed optimization

Francis Bach — Inria - Ecole Normale Supérieure, Paris

Joint work with **K. Scaman**, S. Bubeck, Y.-T. Lee and L. Massoulié

LCCC Workshop - June 2017

# Motivations



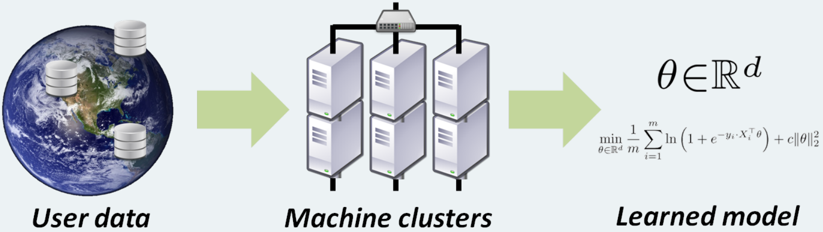

## Typical Machine Learning setting

- Empirical risk minimization:

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^{m} \ell(x_i, y_i; \theta) + c\|\theta\|_2^2$$

- Large scale learning systems handle **massive amounts of data**
- Requires **multiple machines** to train the model

# Motivations




## Typical Machine Learning setting

- Empirical risk minimization: *logistic regression*

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-y_i x_i^\top \theta)) + c\|\theta\|_2^2$$

- Large scale learning systems handle **massive amounts of data**
- Requires **multiple machines** to train the model

# Optimization with a single machine

## "Best" convergence rate for strongly-convex and smooth functions

- Number of iterations to reach a precision $\varepsilon > 0$ (Nesterov, 2004):

$$\Theta\left(\sqrt{\kappa}\ln\left(\frac{1}{\varepsilon}\right)\right)$$

  where $\kappa$ is the condition number of the function to optimize.
- Consequence of $f(\theta_t) - f(\theta^*) \leqslant \beta(1 - 1/\sqrt{\kappa})^t\|\theta_0 - \theta^*\|^2$
- ...but each iteration requires **m gradients to compute**!

# Optimization with a single machine

## "Best" convergence rate for strongly-convex and smooth functions

- Number of iterations to reach a precision $\varepsilon > 0$ (Nesterov, 2004):

$$\Theta\left(\sqrt{\kappa}\ln\left(\frac{1}{\varepsilon}\right)\right)$$

  where $\kappa$ is the condition number of the function to optimize.
- Consequence of $f(\theta_t) - f(\theta^*) \leqslant \beta(1 - 1/\sqrt{\kappa})^t \|\theta_0 - \theta^*\|^2$
- ...but each iteration requires **m gradients to compute**!

## Upper *and* lower bounds of complexity

$$\inf_{\text{algorithms}} \sup_{\text{functions}} \#\text{iterations to reach } \varepsilon$$

- Upper-bound: exhibit an algorithm (here Nesterov acceleration)
- Lower-bound: exhibit a hard function where all algorithms fail

# Distributing information on a network

## Centralized algorithms

- "Master/slave"
- Minimal number of communication steps = Diameter $\Delta$

## Decentralized algorithms

- Gossip algorithms (Boyd et.al., 2006 ; Shah, 2009)
- Mixing time of the Markov chain on the graph $\approx$ inverse of the second smallest eigenvalue $\gamma$ of the Laplacian

# Goals of this work

## Beyond single machine optimization

- Can we improve on $\Theta\left(m\sqrt{\kappa}\ln\left(\frac{1}{\varepsilon}\right)\right)$?
- Is the speed up linear?
- How does a limited bandwidth affects optimization algorithms?

# Goals of this work

## Beyond single machine optimization

- Can we improve on $\Theta\left(m\sqrt{\kappa}\ln\left(\frac{1}{\varepsilon}\right)\right)$?
- Is the speed up linear?
- How does a limited bandwidth affects optimization algorithms?

## Extending optimization theory to distributed architectures

- **Optimal convergence rates** of first order distributed methods,
- **Optimal algorithms** achieving this rate,
- Beyond flat (totally connected) architectures (Arjevani and Shamir, 2015),
- Explicit dependence on optimization parameters and graph parameters.

# Distributed optimization setting

## Optimization problem

Let $f_i$ be $\alpha$-strongly convex and $\beta$-smooth functions. We consider minimizing the average of the local functions.

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

- Machine learning: distributed observations

# Distributed optimization setting

## Optimization problem

Let $f_i$ be $\alpha$-strongly convex and $\beta$-smooth functions. We consider minimizing the average of the local functions.

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

- Machine learning: distributed observations

## Optimization procedures

We consider distributed first-order optimization procedures: access to gradients (or gradients of the Fenchel conjugates).

# Distributed optimization setting

## Optimization problem

Let $f_i$ be $\alpha$-strongly convex and $\beta$-smooth functions. We consider minimizing the average of the local functions.

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

- Machine learning: distributed observations

## Optimization procedures

We consider distributed first-order optimization procedures: access to gradients (or gradients of the Fenchel conjugates).

## Network communications

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a connected simple graph of $n$ computing units and diameter $\Delta$, each having access to a function $f_i(\theta)$ over $\theta \in \mathbb{R}^d$.

# Strong convexity and smoothness

## Strong convexity

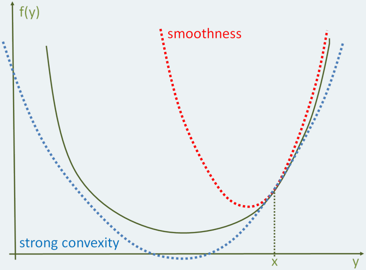
A function $f$ is $\alpha$-strongly convex iff. $\forall x, y \in \mathbb{R}^d$,

$$f(y) \geq f(x) + \nabla f(x)^\top (y-x) + \alpha \|y-x\|^2.$$

## Smoothness

A function $f$ is $\beta$-smooth convex iff. $\forall x, y \in \mathbb{R}^d$,

$$f(y) \leq f(x) + \nabla f(x)^\top (y-x) + \beta \|y-x\|^2.$$



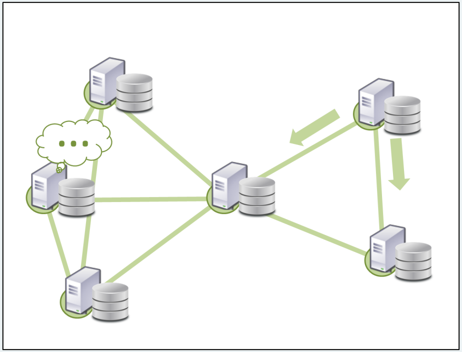

## Notations

- $\kappa_l = \dfrac{\beta}{\alpha}$ (*local*) condition number of each $f_i$,
- $\kappa_g = \dfrac{\beta_g}{\alpha_g}$ (*global*) condition number of $\bar{f}$,
- $\kappa_g \leqslant \kappa_l$, equal if all functions $f_i$ equal.

# Communication network

## Assumptions

- Each **local computation** takes a unit of time,
- Each **communication** between neighbors takes a time $\tau$,
- Actions may be performed in **parallel** and **asynchronously**.



# Distributed optimization algorithms

## Black-box procedures

We consider distributed algorithms verifying the following constraints:

1. **Local memory:** each node $i$ can store past values in an internal memory $\mathcal{M}_{i,t} \subset \mathbb{R}^d$ at time $t \geq 0$.

$$\mathcal{M}_{i,t} \subset \mathcal{M}_{i,t}^{comp} \cup \mathcal{M}_{i,t}^{comm}, \ \theta_{i,t} \in \mathcal{M}_{i,t}.$$

2. **Local computation:** each node $i$ can, at time $t$, compute the gradient of its local function $\nabla f_i(\theta)$ or its Fenchel conjugate $\nabla f_i^*(\theta)$, where $f^*(\theta) = \sup_x x^\top \theta - f(x)$.

$$\mathcal{M}_{i,t}^{comp} = \mathrm{Span}\left(\{\theta, \nabla f_i(\theta), \nabla f_i^*(\theta) : \theta \in \mathcal{M}_{i,t-1}\}\right).$$

3. **Local communication:** each node $i$ can, at time $t$, share a value to all or part of its neighbors.

$$\mathcal{M}_{i,t}^{comm} = \mathrm{Span}\left(\bigcup_{(i,j)\in\mathcal{E}} \mathcal{M}_{j,t-\tau}\right).$$

# Centralized vs. decentralized architectures

## Centralized communication

- One **master** machine is responsible for sending requests and synchronizing computation,
- **Slave** machines perform computations upon request and send the result to the master.

# Centralized vs. decentralized architectures

## Centralized communication

- One **master** machine is responsible for sending requests and synchronizing computation,
- **Slave** machines perform computations upon request and send the result to the master.

## Decentralized communication

- All machines perform local computations and share values with their neighbors,
- Local averaging is performed through **gossip** (Boyd et.al., 2006).
- Node $i$ receives $\sum_j W_{ij} x_j = (Wx)_i$, where $W$ verifies:
  1. $W$ is an $n \times n$ symmetric matrix,
  2. $W$ is defined on the edges of the network: $W_{ij} \neq 0$ only if $i = j$ or $(i, j) \in \mathcal{E}$,
  3. $W$ is positive semi-definite,
  4. The kernel of $W$ is the set of constant vectors: $\mathrm{Ker}(W) = \mathrm{Span}(\mathbb{1})$, where $\mathbb{1} = (1, ..., 1)^\top$.
- Let $\gamma(W) = \lambda_{n-1}(W)/\lambda_1(W)$ be the (normalized) eigengap of $W$.

# Lower bound on convergence rate

## Theorem 1 (SBBLM, 2017)

Let $\mathcal{G}$ be a graph of diameter $\Delta > 0$ and size $n > 0$, and $\beta_g \geq \alpha_g > 0$. There exist $n$ functions $f_i : \ell_2 \to \mathbb{R}$ such that $\bar{f}$ is $\alpha_g$-strongly-convex and $\beta_g$-smooth, and for any $t \geq 0$ and any black-box procedure one has, for all $i \in \{1, ..., n\}$,

$$\bar{f}(\theta_{i,t}) - \bar{f}(\theta^*) \geq \frac{\alpha_g}{2} \left(1 - \frac{4}{\sqrt{\kappa_g}}\right)^{1 + \frac{t}{1 + \Delta\tau}} \|\theta_{i,0} - \theta^*\|^2.$$

# Lower bound on convergence rate

## Theorem 1 (SBBLM, 2017)

Let $\mathcal{G}$ be a graph of diameter $\Delta > 0$ and size $n > 0$, and $\beta_g \geq \alpha_g > 0$. There exist $n$ functions $f_i : \ell_2 \to \mathbb{R}$ such that $\bar{f}$ is $\alpha_g$-strongly-convex and $\beta_g$-smooth, and for any $t \geq 0$ and any black-box procedure one has, for all $i \in \{1, ..., n\}$,

$$\bar{f}(\theta_{i,t}) - \bar{f}(\theta^*) \geq \frac{\alpha_g}{2}\left(1 - \frac{4}{\sqrt{\kappa_g}}\right)^{1+\frac{t}{1+\Delta\tau}} \|\theta_{i,0} - \theta^*\|^2.$$

## Take-home message

For any graph of diameter $\Delta$ and any black-box procedure, there exist functions $f_i$ such that the time to reach a precision $\varepsilon > 0$ is lower bounded by

$$\boxed{\Omega\left(\sqrt{\kappa_g}\left(1 + \Delta\tau\right)\ln\left(\frac{1}{\varepsilon}\right)\right)}$$

- Extends the totally connected result of Arjevani & Shamir (2015)

# Proof warm-up: single machine

- **Simplification:** $\ell_2$ instead of $\mathbb{R}^d$.
- **Goal**: design a worst-case convex function $f$.
- From Nesterov (2004), Bubeck (2015):

$$f(\theta) = \frac{\alpha(\kappa - 1)}{8}\left[\theta^\top A\theta - 2\theta_1\right] + \frac{\alpha}{2}\|\theta\|_2^2$$

with $A$ infinite tridiagonal matrix with 2 on the diagonal, and $-1$ on the upper and lower diagonal.

# Proof warm-up: single machine

- **Simplification:** $\ell_2$ instead of $\mathbb{R}^d$.
- **Goal**: design a worst-case convex function $f$.
- From Nesterov (2004), Bubeck (2015):

$$f(\theta) = \frac{\alpha(\kappa - 1)}{8}\big[\theta^\top A\theta - 2\theta_1\big] + \frac{\alpha}{2}\|\theta\|_2^2$$

  with $A$ infinite tridiagonal matrix with 2 on the diagonal, and $-1$ on the upper and lower diagonal.
- **Facts 1**: $0 \preccurlyeq A \preccurlyeq 4I$, $f$ is $\alpha$-strongly convex and $\beta$-smooth
- **Fact 2**: starting from $\theta_0 = 0$, after $t$ gradient steps, $\theta_t$ is supported on the first $t$ coordinates $\Rightarrow \|\theta_t - \theta^*\|^2 \geqslant \sum_{i>t}\|\theta_i^*\|^2$
- Get lower bound $f(\theta_t) - f(\theta^*) \geqslant \frac{\alpha}{2}\big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\big)^{2t}\|\theta_0 - \theta_*\|^2$ after some computations

# Proof warm-up: single machine

- **Simplification:** $\ell_2$ instead of $\mathbb{R}^d$.
- **Goal**: design a worst-case convex function $f$.
- From Nesterov (2004), Bubeck (2015):

$$f(\theta) = \frac{\alpha(\kappa-1)}{8}\left[\theta^\top A\theta - 2\theta_1\right] + \frac{\alpha}{2}\|\theta\|_2^2$$

  with $A$ infinite tridiagonal matrix with 2 on the diagonal, and $-1$ on the upper and lower diagonal. $\theta^\top A\theta = \theta_1^2 + \sum_{i\geqslant 1}(\theta_i - \theta_{i+1})^2$
- **Facts 1**: $0 \preccurlyeq A \preccurlyeq 4I$, $f$ is $\alpha$-strongly convex and $\beta$-smooth
- **Fact 2**: starting from $\theta_0 = 0$, after $t$ gradient steps, $\theta_t$ is supported on the first $t$ coordinates $\Rightarrow \|\theta_t - \theta^*\|^2 \geqslant \sum_{i>t}\|\theta_i^*\|^2$
- Get lower bound $f(\theta_t) - f(\theta^*) \geqslant \frac{\alpha}{2}\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{2t}\|\theta_0 - \theta_*\|^2$ after some computations

# Proof sketch (1)

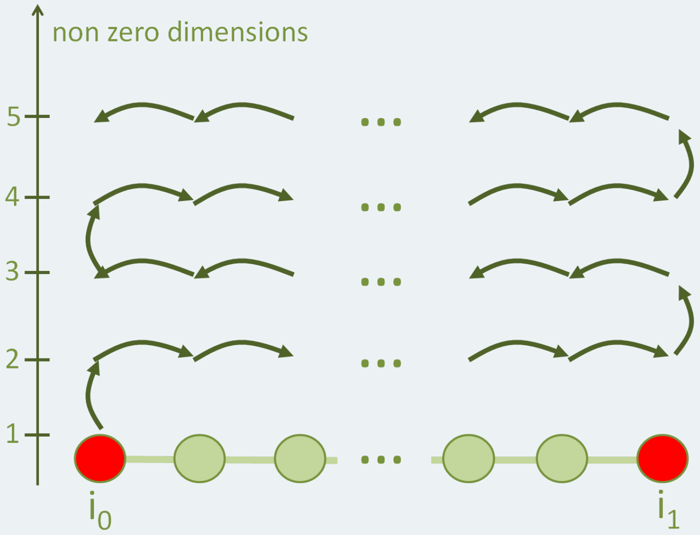
- **Simplification:** $\ell_2$ instead of $\mathbb{R}^d$.
- **Extremal nodes:** $i_0$ and $i_1$ at distance $\Delta$.
- **Functions to optimize:** Splitting the usual Nesterov function

$$f_i(\theta) = \begin{cases} \frac{\alpha}{2}\|\theta\|_2^2 + n\frac{\beta-\alpha}{8}(\theta^\top M_1\theta - \theta_1) & \text{if } i = i_0 \\ \frac{\alpha}{2}\|\theta\|_2^2 + n\frac{\beta-\alpha}{8}\theta^\top M_2\theta & \text{if } i = i_1 \\ \frac{\alpha}{2}\|\theta\|_2^2 & \text{otherwise} \end{cases}$$

where $M_1 : \ell_2 \to \ell_2$ is the infinite block diagonal matrix with $\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$ on the diagonal, and $M_2 = \begin{pmatrix} 1 & 0 \\ 0 & M_1 \end{pmatrix}$.

- **Optimal value:** $\theta_k^* = \left(\frac{\sqrt{\beta}-\sqrt{\alpha}}{\sqrt{\beta}+\sqrt{\alpha}}\right)^k$.

# Proof sketch (2)



# Proof sketch (3)

- If $\theta_{i,0} = 0$, each local computation can only **increase the number of non zero dimensions by one**.
- $\nabla f_{i_0}(\theta_{i_0,t})$ increases **odd** dimensions, $\nabla f_{i_1}(\theta_{i_1,t})$ increases **even** dimensions.
- $\Delta$ communication steps are required to communicate between $i_0$ and $i_1$.
- $\theta_{i,t,k} \neq 0$ after at least $k$ computation steps and $k\Delta$ communication steps.
- $\bar{f}$ is $\alpha$-strongly convex and $\beta$-smooth, and

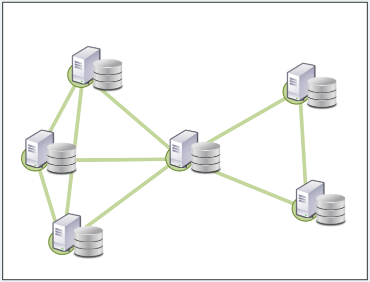
$$\bar{f}(\theta_{i,t}) - \bar{f}(\theta^*) \geq \frac{\alpha}{2}\|\theta_{i,t} - \theta^*\|_2^2 \geq \frac{\alpha}{2} \sum_{k=k_{i,t}+1}^{+\infty} {\theta_k^*}^2,$$

where $k_{i,t} = \max\{k \in \mathbb{N} \ : \ \exists \theta \in \mathcal{M}_{i,t} \text{ s.t. } \theta_k \neq 0\} \leq \left\lfloor \frac{t+\Delta\tau}{1+\Delta\tau} \right\rfloor$.

# Simple is good...!

## Master/slave algorithm

Simple master/slave distribution of Nesterov's accelerated gradient descent.



**Input:** number of iterations $T > 0$, communication network $\mathcal{G}$, $\eta = \frac{1}{\beta_g}$, $\mu = \frac{\sqrt{\kappa_g}-1}{\sqrt{\kappa_g}+1}$

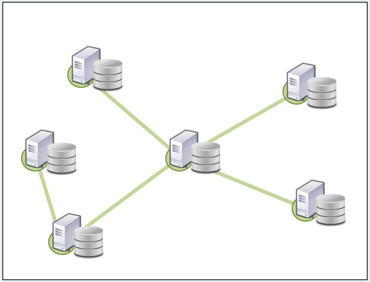
**Output:** $\theta_T$

Compute a spanning tree $\mathcal{T}$ on $\mathcal{G}$
$\theta_0 = 0$, $y_0 = 0$
**for** $t = 0$ to $T-1$ **do**
    Send $\theta_t$ to all nodes through $\mathcal{T}$
    $\nabla \bar{f}(\theta_t) = \textsc{aggregateGradients}(\theta_t)$
    $y_{t+1} = \theta_t - \eta \nabla \bar{f}(\theta_t)$
    $\theta_{t+1} = (1+\mu) y_{t+1} - \mu y_t$
**end for**
**return** $\theta_T$

# Simple is good...!

## Master/slave algorithm

Simple master/slave distribution of Nesterov's accelerated gradient descent.



**Input:** number of iterations $T > 0$, communication network $\mathcal{G}$, $\eta = \frac{1}{\beta_g}$, $\mu = \frac{\sqrt{\kappa_g}-1}{\sqrt{\kappa_g}+1}$

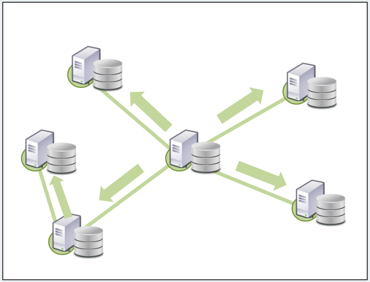
**Output:** $\theta_T$

1: Compute a spanning tree $\mathcal{T}$ on $\mathcal{G}$
2: $\theta_0 = 0$, $y_0 = 0$
3: **for** $t = 0$ to $T-1$ **do**
4: Send $\theta_t$ to all nodes through $\mathcal{T}$
5: $\nabla \bar{f}(\theta_t) = \textsc{aggregateGradients}(\theta_t)$
6: $y_{t+1} = \theta_t - \eta \nabla \bar{f}(\theta_t)$
7: $\theta_{t+1} = (1+\mu) y_{t+1} - \mu y_t$
8: **end for**
9: **return** $\theta_T$

# Simple is good...!

## Master/slave algorithm

Simple master/slave distribution of Nesterov's accelerated gradient descent.



**Input:** number of iterations $T > 0$,
communication network $\mathcal{G}$, $\eta = \frac{1}{\beta_g}$,
$\mu = \frac{\sqrt{\kappa_g}-1}{\sqrt{\kappa_g}+1}$

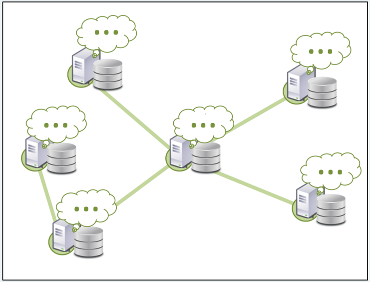
**Output:** $\theta_T$

Compute a spanning tree $\mathcal{T}$ on $\mathcal{G}$
$\theta_0 = 0$, $y_0 = 0$
**for** $t = 0$ to $T - 1$ **do**
  Send $\theta_t$ to all nodes through $\mathcal{T}$
  $\nabla \bar{f}(\theta_t) = \textsc{aggregateGradients}(\theta_t)$
  $y_{t+1} = \theta_t - \eta \nabla \bar{f}(\theta_t)$
  $\theta_{t+1} = (1 + \mu) y_{t+1} - \mu y_t$
**end for**
**return** $\theta_T$

# Simple is good...!

## Master/slave algorithm

Simple master/slave distribution of Nesterov's accelerated gradient descent.



**Input:** number of iterations $T > 0$, communication network $\mathcal{G}$, $\eta = \frac{1}{\beta_g}$, $\mu = \frac{\sqrt{\kappa_g}-1}{\sqrt{\kappa_g}+1}$

**Output:** $\theta_T$

Compute a spanning tree $\mathcal{T}$ on $\mathcal{G}$

$\theta_0 = 0$, $y_0 = 0$

**for** $t = 0$ to $T-1$ **do**

  Send $\theta_t$ to all nodes through $\mathcal{T}$

  $\nabla \bar{f}(\theta_t) = \textsc{aggregateGradients}(\theta_t)$

  $y_{t+1} = \theta_t - \eta \nabla \bar{f}(\theta_t)$

  $\theta_{t+1} = (1+\mu) y_{t+1} - \mu y_t$

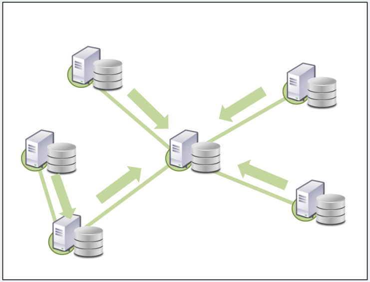
**end for**

**return** $\theta_T$

# Simple is good...!

## Master/slave algorithm

Simple master/slave distribution of Nesterov's accelerated gradient descent.



**Input:** number of iterations $T > 0$, communication network $\mathcal{G}$, $\eta = \frac{1}{\beta_g}$, $\mu = \frac{\sqrt{\kappa_g}-1}{\sqrt{\kappa_g}+1}$

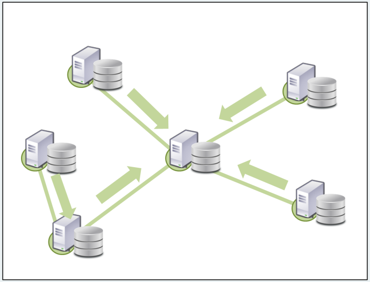
**Output:** $\theta_T$

Compute a spanning tree $\mathcal{T}$ on $\mathcal{G}$
$\theta_0 = 0$, $y_0 = 0$
**for** $t = 0$ to $T-1$ **do**
    Send $\theta_t$ to all nodes through $\mathcal{T}$
    $\nabla \bar{f}(\theta_t) = \textsc{aggregateGradients}(\theta_t)$
    $y_{t+1} = \theta_t - \eta \nabla \bar{f}(\theta_t)$
    $\theta_{t+1} = (1+\mu) y_{t+1} - \mu y_t$
**end for**
**return** $\theta_T$

# Simple is good...!

## Master/slave algorithm

Simple master/slave distribution of Nesterov's accelerated gradient descent.



## Convergence rate

- Each iteration requires a time $1 + 2\Delta\tau$,
- Reaches a precision $\varepsilon > 0$ in time

$$O\left(\sqrt{\kappa_g}\left(1+\Delta\tau\right)\ln\left(\frac{1}{\varepsilon}\right)\right).$$

# Drawbacks

## Drawbacks of this approach

- Not robust to changes in the connectivity of the network,
- Requires waiting for all machines to compute their local gradients.

# Drawbacks

## Drawbacks of this approach

- Not robust to changes in the connectivity of the network,
- Requires waiting for all machines to compute their local gradients.

## A natural solution: decentralized algorithms

- Asynchronous computations,
- Machines do not wait for one another,
- Communication is not interrupted by a change in the network.

# Related works

## Large literature for decentralized optimization

- Distributed SGD (Nedic & Ozdaglar, 2009) $O(\frac{n^3 R^2 L^2}{\varepsilon^2})$
- Decentralized dual averaging (Duchi et al., 2012) $O(\frac{R^2 L^2}{\gamma(W)\varepsilon^2})$
- D-ADMM (Boyd et al., 2011; Wei & Ozdaglar, 2012; Shi et al., 2014 ; Lutzeler et al., 2016) $O(\frac{2\kappa_l^2}{\sqrt{1+4\kappa_l^2\gamma(W)}-1}\ln\left(\frac{1}{\varepsilon}\right))$
- EXTRA algorithm (Shi et al., 2015; Mokhtari & Ribeiro, 2016) $\exists\delta > 0$ s.t. $O(\delta\ln\left(\frac{1}{\varepsilon}\right))$
- Augmented Lagrangians (Jakovetić et al., 2015) $O(\frac{2\kappa_l^2}{\sqrt{1+4\kappa_l^2\gamma(W)}-1}\ln\left(\frac{1}{\varepsilon}\right))$
- DIGing (Nedich et al., 2016) $O(n^{4.5}\kappa_l^{1.5}\ln\left(\frac{1}{\varepsilon}\right))$
- ...

# Decentralized algorithms

## Optimal convergence rate?

- Decentralized convergence rates usually depend on the (normalized) eigengap $\gamma(W)$,
- For simple graphs (linear graphs, regular graphs), $\Delta \approx \frac{1}{\sqrt{\gamma(W)}}$, where $W$ is the Laplacian matrix,
- Can we have $\Theta\left(\sqrt{\kappa_g}\left(1 + \frac{\tau}{\sqrt{\gamma(W)}}\right)\ln\left(\frac{1}{\varepsilon}\right)\right)$?

# Decentralized algorithms

## Optimal convergence rate?

- Decentralized convergence rates usually depend on the (normalized) eigengap $\gamma(W)$,
- For simple graphs (linear graphs, regular graphs), $\Delta \approx \frac{1}{\sqrt{\gamma(W)}}$, where $W$ is the Laplacian matrix,
- Can we have $\Theta\left(\sqrt{\kappa_g}\left(1+\frac{\tau}{\sqrt{\gamma(W)}}\right)\ln\left(\frac{1}{\varepsilon}\right)\right)$?
- No! Sometimes $\frac{1}{\sqrt{\gamma(W)}} \approx \frac{\Delta}{\ln n}$ (Ramanujan graphs and Erdös-Rényi random networks)...

# Decentralized algorithms

## Optimal convergence rate?

- Decentralized convergence rates usually depend on the (normalized) eigengap $\gamma(W)$,
- For simple graphs (linear graphs, regular graphs), $\Delta \approx \frac{1}{\sqrt{\gamma(W)}}$, where $W$ is the Laplacian matrix,
- Can we have $\Theta\left(\sqrt{\kappa_g}\left(1 + \frac{\tau}{\sqrt{\gamma(W)}}\right)\ln\left(\frac{1}{\varepsilon}\right)\right)$?
- No! Sometimes $\frac{1}{\sqrt{\gamma(W)}} \approx \frac{\Delta}{\ln n}$ (Ramanujan graphs and Erdös-Rényi random networks)...

## Optimal algorithm?

- We can achieve this rate if we replace $\kappa_g$ by $\kappa_l \geq \kappa_g$,
- Based on a **double acceleration**: accelerated gradient descent **and** accelerated gossip!

# Lower bound on convergence rate

## Theorem 2 (SBBLM, 2017)

Let $\alpha, \beta > 0$ and $\gamma \in (0, 1]$. There exists a gossip matrix $W$ of eigengap $\gamma(W) = \gamma$, and $\alpha$-strongly convex and $\beta$-smooth functions $f_i : \ell_2 \to \mathbb{R}$ such that, for any $t \geq 0$ and any black-box procedure using $W$ one has, for all $i \in \{1, ..., n\}$,

$$\bar{f}(\theta_{i,t}) - \bar{f}(\theta^*) \geq \frac{3\alpha}{2} \left(1 - \frac{16}{\sqrt{\kappa_l}}\right)^{1+\frac{t}{1+\frac{\tau}{5\sqrt{\gamma}}}} \|\theta_{i,0} - \theta^*\|^2.$$

# Lower bound on convergence rate

## Theorem 2 (SBBLM, 2017)

Let $\alpha, \beta > 0$ and $\gamma \in (0, 1]$. There exists a gossip matrix $W$ of eigengap $\gamma(W) = \gamma$, and $\alpha$-strongly convex and $\beta$-smooth functions $f_i : \ell_2 \to \mathbb{R}$ such that, for any $t \geq 0$ and any black-box procedure using $W$ one has, for all $i \in \{1, ..., n\}$,

$$\bar{f}(\theta_{i,t}) - \bar{f}(\theta^*) \geq \frac{3\alpha}{2}\left(1 - \frac{16}{\sqrt{\kappa_l}}\right)^{1+\frac{t}{1+\frac{\tau}{5\sqrt{\gamma}}}} \|\theta_{i,0} - \theta^*\|^2.$$

## Take-home message

For any $\gamma > 0$, there exists a gossip matrix $W$ of eigengap $\gamma$ there exist functions $f_i$ such that the time to reach a precision $\varepsilon > 0$ is lower bounded by

$$\boxed{\Omega\left(\sqrt{\kappa_l}\left(1 + \frac{\tau}{\sqrt{\gamma}}\right)\ln\left(\frac{1}{\varepsilon}\right)\right)}$$

# Lower bound on convergence rate

## Theorem 2 (SBBLM, 2017)

Let $\alpha, \beta > 0$ and $\gamma \in (0, 1]$. There exists a gossip matrix $W$ of eigengap $\gamma(W) = \gamma$, and $\alpha$-strongly convex and $\beta$-smooth functions $f_i : \ell_2 \to \mathbb{R}$ such that, for any $t \geq 0$ and any black-box procedure using $W$ one has, for all $i \in \{1, ..., n\}$,

$$\bar{f}(\theta_{i,t}) - \bar{f}(\theta^*) \geq \frac{3\alpha}{2}\left(1 - \frac{16}{\sqrt{\kappa_l}}\right)^{1 + \frac{t}{1 + \frac{\tau}{5\sqrt{\gamma}}}} \|\theta_{i,0} - \theta^*\|^2.$$

## Take-home message

For any $\gamma > 0$, there exists a gossip matrix $W$ of eigengap $\gamma$ there exist functions $f_i$ such that the time to reach a precision $\varepsilon > 0$ is lower bounded by

$$\boxed{\Omega\left(\sqrt{\kappa_l}\left(1 + \frac{\tau}{\sqrt{\gamma}}\right) \ln\left(\frac{1}{\varepsilon}\right)\right)}$$

## Naive algorithm does not work!

# Reformulation of the optimization problem

- Using the gossip matrix to ensure equality of all $\theta_i$ (Jakovetić et al., 2015),

$$\min_{\theta \in \mathbb{R}^d} \bar{f}(\theta) = \min_{\Theta \in \mathbb{R}^{d \times n} \, : \, \Theta \sqrt{W} = 0} F(\Theta),$$

where $F(\Theta) = \sum_{i=1}^n f_i(\theta_i)$, with $\Theta = (\theta_1, \dots, \theta_n) \in \mathbb{R}^{d \times n}$

# Reformulation of the optimization problem

- Using the gossip matrix to ensure equality of all $\theta_i$ (Jakovetić et al., 2015),

$$\min_{\theta \in \mathbb{R}^d} \bar{f}(\theta) = \min_{\Theta \in \mathbb{R}^{d \times n} \,:\, \Theta\sqrt{W}=0} F(\Theta),$$

where $F(\Theta) = \sum_{i=1}^n f_i(\theta_i)$, with $\Theta = (\theta_1, \dots, \theta_n) \in \mathbb{R}^{d \times n}$

- Dual version:

$$\boxed{\max_{\lambda \in \mathbb{R}^{d \times n}} -F^*(\lambda\sqrt{W})}$$

- Gradient descent in the dual:

$$\lambda_{t+1} = \lambda_t - \eta \nabla F^*(\lambda_t \sqrt{W})\sqrt{W},$$

and the change of variable $y_t = \lambda_t \sqrt{W}$ leads to

$$y_{t+1} = y_t - \eta \nabla F^*(y_t) W.$$

# A double acceleration: (1) accelerated gradient descent

- The dual problem
$$\max_{\lambda \in \mathbb{R}^{d \times n}} -F^*(\lambda\sqrt{W})$$
is an unconstrained strongly convex and smooth problem with condition number $\frac{\kappa_l}{\gamma(W)}$.

# A double acceleration: (1) accelerated gradient descent

- The dual problem
$$\max_{\lambda \in \mathbb{R}^{d \times n}} -F^*(\lambda \sqrt{W})$$
is an unconstrained strongly convex and smooth problem with condition number $\frac{\kappa_l}{\gamma(W)}$.
- Nesterov's accelerated gradient descent reaches a precision $\varepsilon > 0$ in
$$O\left(\sqrt{\frac{\kappa_l}{\gamma(W)}}\,(1+\tau)\ln\left(\frac{1}{\varepsilon}\right)\right).$$
- Optimal w.r.t. the communication time... but not in the number of gradient steps.

# A double acceleration: (2) accelerated gossip

- Only one gossip step per local computation: suboptimal when $\tau \ll 1$!
- Accelerated gossip: replacing $W$ by a polynomial $P_K(W)$.
  - Cao et al. (2006), Kokiopoulou and Frossard (2009), Cavalcante et al. (2011)

# A double acceleration: (2) accelerated gossip

- Only one gossip step per local computation: suboptimal when $\tau \ll 1$!
- Accelerated gossip: replacing $W$ by a polynomial $P_K(W)$.
  - Cao et al. (2006), Kokiopoulou and Frossard (2009), Cavalcante et al. (2011)
- **Chebyshev polynomials** lead to the best convergence rates:

$$P_K(x) = 1 - \frac{T_K(c_2(1-x))}{T_K(c_2)},$$

where $c_2 = \frac{1+\gamma}{1-\gamma}$ and $T_K$ are the Chebyshev polynomials defined as $T_0(x) = 1,\ T_1(x) = x$, and, for all $k \geq 1$,

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x).$$

# A double acceleration: (2) accelerated gossip

- Only one gossip step per local computation: suboptimal when $\tau \ll 1$!
- Accelerated gossip: replacing $W$ by a polynomial $P_K(W)$.
  - Cao et al. (2006), Kokiopoulou and Frossard (2009), Cavalcante et al. (2011)
- **Chebyshev polynomials** lead to the best convergence rates:

$$P_K(x) = 1 - \frac{T_K(c_2(1-x))}{T_K(c_2)},$$

where $c_2 = \frac{1+\gamma}{1-\gamma}$ and $T_K$ are the Chebyshev polynomials defined as $T_0(x) = 1,\ T_1(x) = x$, and, for all $k \geq 1$,

$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x).$$

- With $K = \left\lfloor \frac{1}{\sqrt{\gamma(W)}} \right\rfloor$, reaches a precision $\varepsilon > 0$ in time

$$O\left(\sqrt{\frac{\kappa_l}{\gamma(P_K(W))}}(1 + K\tau)\ln\left(\frac{1}{\varepsilon}\right)\right) = O\left(\sqrt{\kappa_l}\left(1 + \frac{\tau}{\sqrt{\gamma}}\right)\ln\left(\frac{1}{\varepsilon}\right)\right).$$

# Optimal decentralized algorithm

## Multi-step Dual Accelerated (MSDA)

**Input:** gossip matrix $W \in \mathbb{R}^{n \times n}$, $T > 0$
**Output:** $\theta_{i,T}$, for $i = 1, \dots, n$

$x_0 = 0$, $y_0 = 0$
**for** $t = 0$ to $T - 1$ **do**
  $\theta_{i,t} = \nabla f_i^*(x_{i,t})$, for all $i = 1, \dots, n$
  $y_{t+1} = x_t - \eta\, \textsc{accGossip}(\Theta_t, W, K)$
  $x_{t+1} = (1 + \mu) y_{t+1} - \mu y_t$
**end for**

**procedure** $\textsc{accGossip}(x, W, K)$
$a_0 = 1$, $a_1 = c_2$
$x_0 = x$, $x_1 = c_2 x(I - c_3 W)$
**for** $k = 1$ to $K - 1$ **do**
  $a_{k+1} = 2c_2 a_k - a_{k-1}$
  $x_{k+1} = 2c_2 x_k (I - c_3 W) - x_{k-1}$
**end for**
**return** $x_0 - \frac{x_K}{a_K}$
**end procedure**

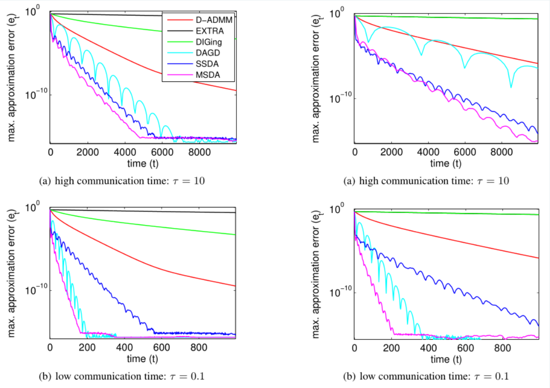
# Experiments: logistic regression

## Optimization problem

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^{m} \ln\left(1 + e^{-y_i \cdot x_i^\top \theta}\right) + c\|\theta\|_2^2$$

## Communication network

- Left: Erdös-Rényi random graph of 100 nodes and average degree 6,
- Right: Square grid of $10 \times 10$ nodes.




(a) high communication time: $\tau = 10$ (a) high communication time: $\tau = 10$

(b) low communication time: $\tau = 0.1$ (b) low communication time: $\tau = 0.1$

## Conclusion

- First optimal convergence rates for distrbuted optimization in networks,
- Optimal centralized convergence rate: $\Theta\left(\sqrt{\kappa_g}\left(1 + \Delta\tau\right) \ln\left(\frac{1}{\varepsilon}\right)\right)$,
- Optimal decentralized convergence rate: $\Theta\left(\sqrt{\kappa_l}\left(1 + \frac{\tau}{\sqrt{\gamma}}\right) \ln\left(\frac{1}{\varepsilon}\right)\right)$.

## Conclusion

- First optimal convergence rates for distrbuted optimization in networks,
- Optimal centralized convergence rate: $\Theta\left(\sqrt{\kappa_g}\left(1+\Delta\tau\right)\ln\left(\frac{1}{\varepsilon}\right)\right)$,
- Optimal decentralized convergence rate: $\Theta\left(\sqrt{\kappa_l}\left(1+\frac{\tau}{\sqrt{\gamma}}\right)\ln\left(\frac{1}{\varepsilon}\right)\right)$.

## Extensions

- Beyond strong-convexity, stochastic problems
- Asynchronous algorithms
- Decentralized rate in $\kappa_g$?
- Primal-only optimal decentralized algorithm,
  - Composite functions $f_i(\theta) = g_i(B_i\theta) + c\|\theta\|^2$
  - Approximation of the proximal point algorithm
- Time varying networks, delays, failures, etc.

Thank you!