Theoretical foundations towards practical decentralized optimization

...

Hadrien Hendrikx, Inria Thoth

Core-AI Seminar, Montpellier - April 9, 2026

I - An introduction to distributed learning

Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



Distributed Optimization?

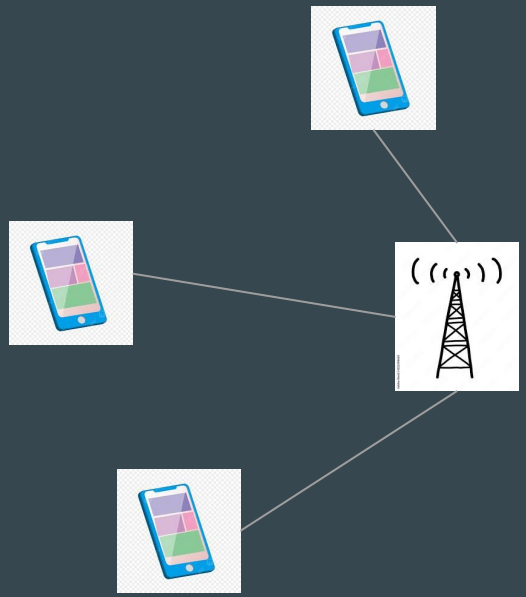
How to perform Matrix-vector multiplication in parallel ?

- You can parallelize over the cores of your computer
- You can give different blocks to different computers

The difference: how expensive is it to share information?

=> Distributed optimization grants you access to more resources. The important metric is scaling : what's the speedup depending on the number of cores/computers you use?

The setting: distributed not parallel



- Several agents: the phones
- A common task:
  - Next word prediction for keyboards
  - Plants identification based on photos
  - Channel estimation
- A communication network

Federated Learning?

Data is produced by different nodes, and stays there:

- Because of privacy reasons (don't want to share your data)
- Because of cost reasons (too big to transfer)

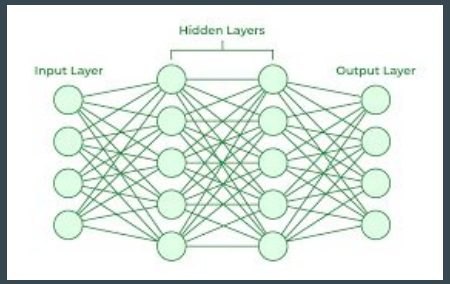
Therefore, nodes HAVE to communicate in order to minimize a shared objective.
Scaling is important, but other troubles arise:

- Communications are assumed to be expensive
- Nodes might not always be connected
- Some nodes might behave adversarially

Learning comes down to minimizing losses

Image x





$p(\theta, x)$

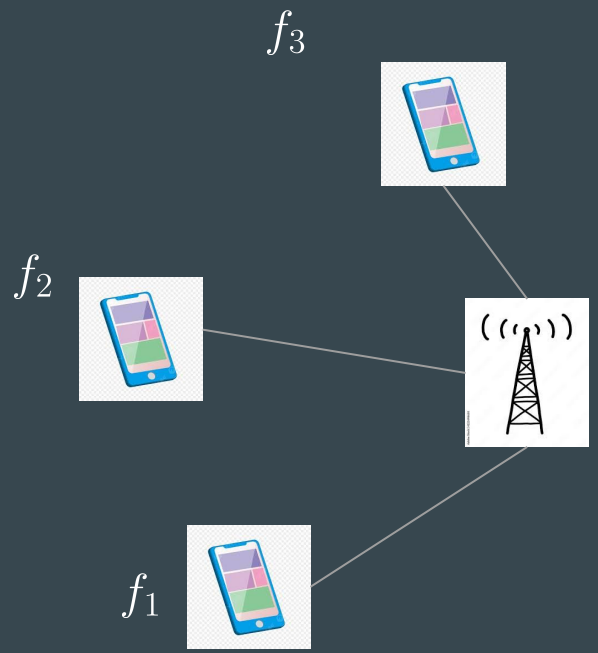
Edible probability

Model weights θ

Label $y = 1$ (edible)

Loss $$f_i(\theta) = \sum_{x,y \in D_i} \|p(\theta, x) - y\|^2$$

Distributed Learning



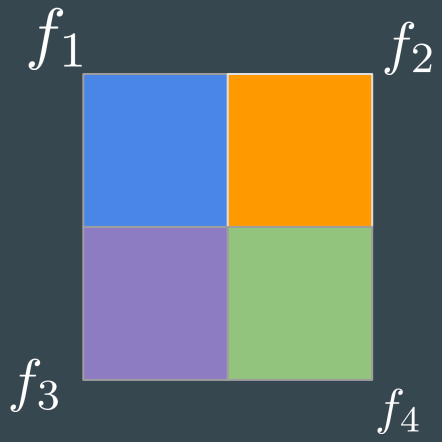
A common task:

minimize the average of local errors

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} f_i(\theta)$$

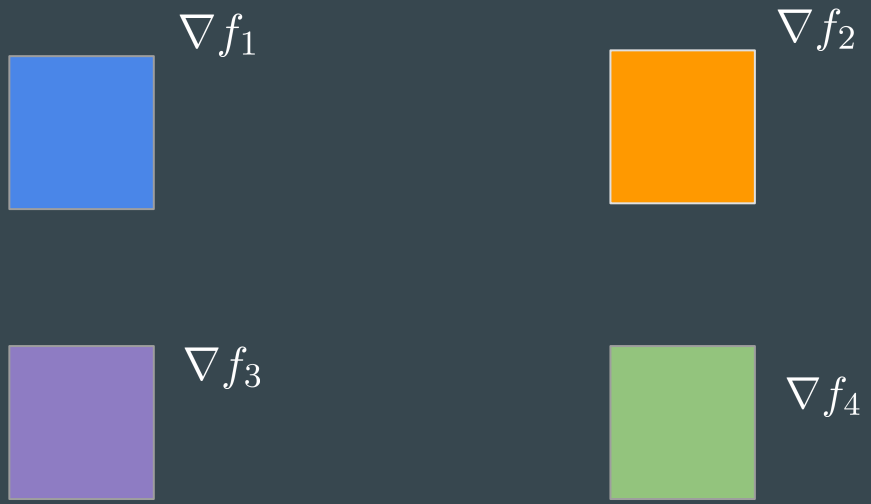
Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



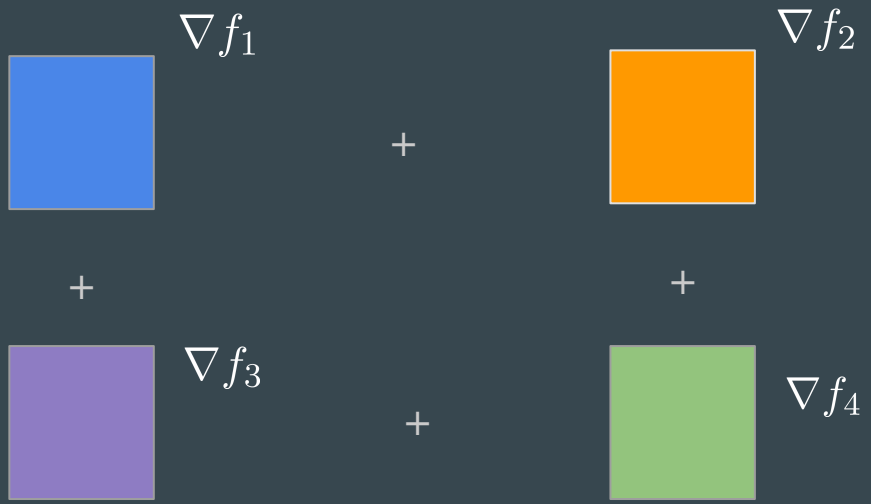
Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



Distributed Optimization?

How to perform Matrix-vector multiplication in parallel ?



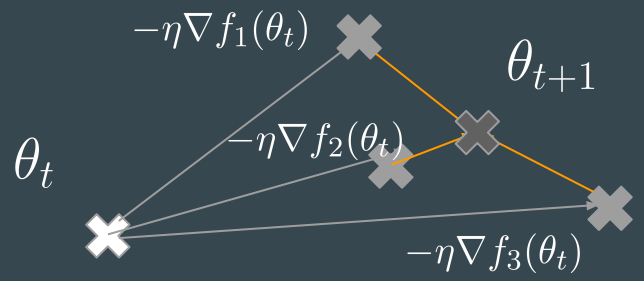
Gradient Descent

In machine learning, these problems are generally solved using gradient methods.

- If all the data were in one place: $\theta_{t+1} = \theta_t - \eta \nabla \left(\frac{1}{n} \sum_{i=1}^{n} f_i(\theta) \right)$
- What we do instead: $\theta_{t+1} = \frac{1}{n} \sum_{i=1}^{n} (\theta_t - \eta \nabla f_i(\theta_t))$

Update locally with (stochastic) gradients: requires compute
then average globally: requires communication

How do people do this in practice ?



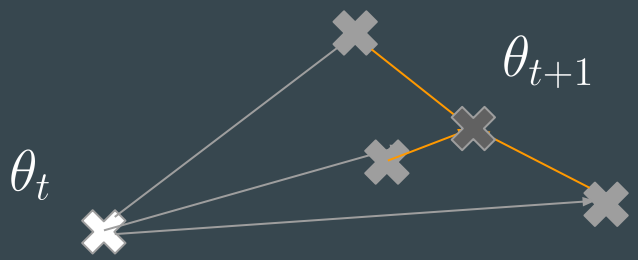
Gradient step increases the distance between points and their average

Averaging step brings points back together

Completely equivalent to the single-machine algorithm:

- “easy” to analyze
- Quite efficient

How do people do this in practice ?



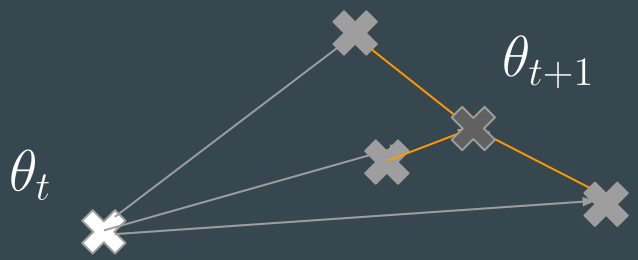
Gradient step increases the distance between points and their average

Averaging step brings points back together

SGD, the most common algorithm, relies on cheap updates

Many averaging steps (and so, communications) are required. We would like to compute more between communications.

How do people do this in practice ?



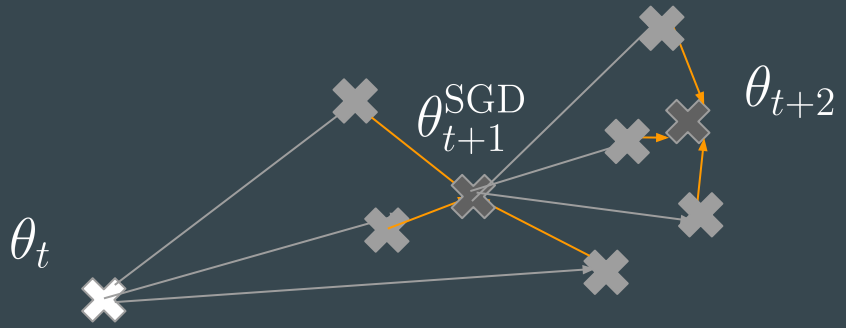
Gradient step increases the distance between points and their average

Averaging step brings points back together

Many averaging steps (and so, communications) are required. We would like to compute more between communications.

- Compute gradients on more data: moderate scaling
- Perform multiple update steps between each communication step

Distributed SGD

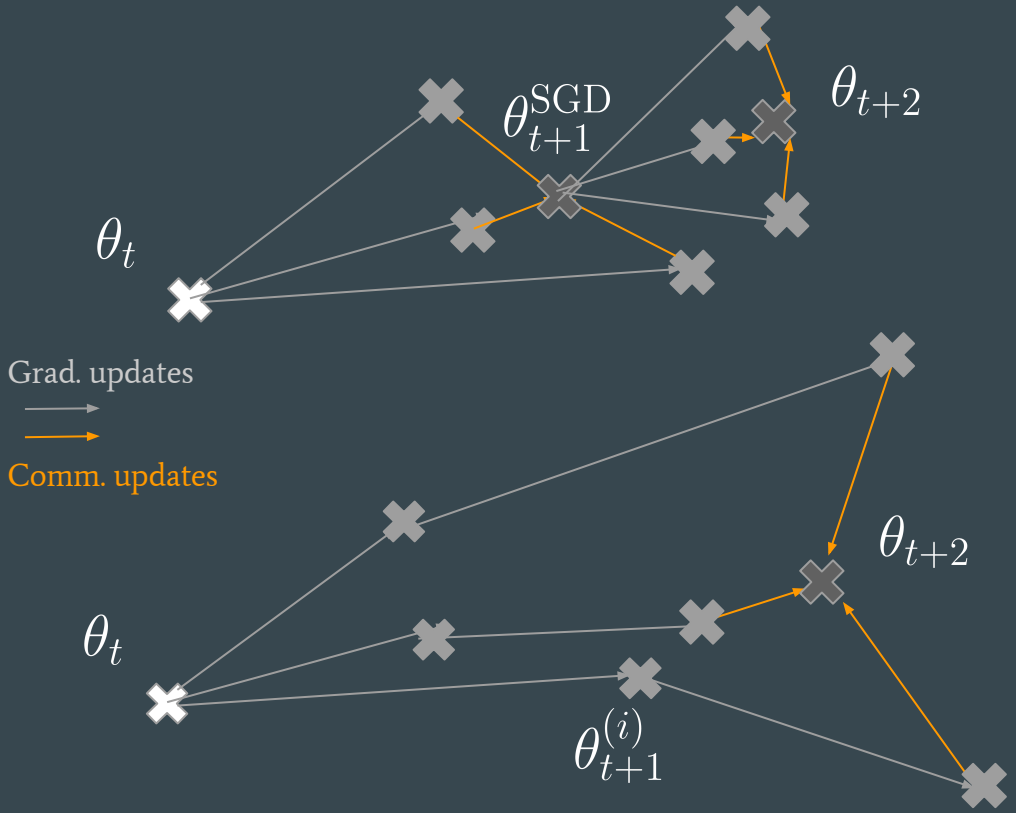


Grad. updates

Comm. updates

SGD: communication after each stochastic gradient update

Federated Averaging



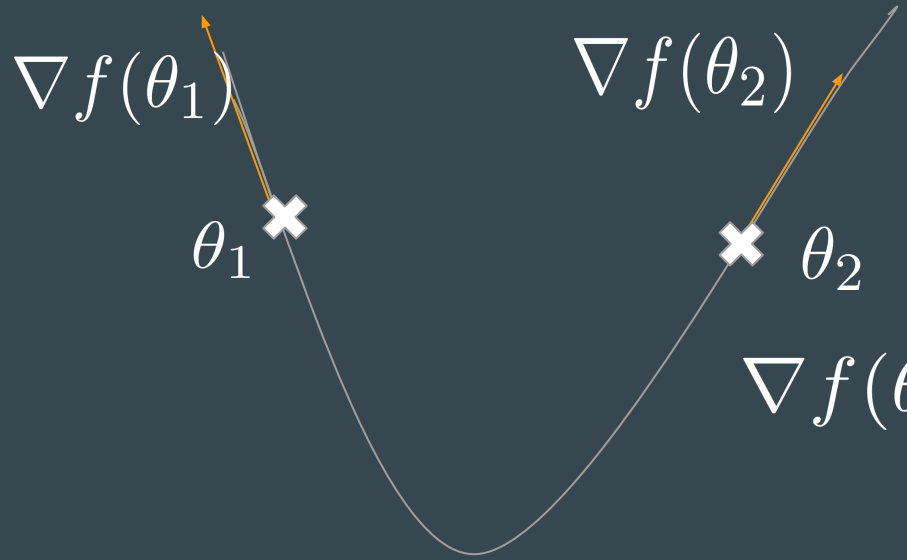
SGD: communication after each stochastic gradient update

FedAvg: Communication only after a few local update steps.

$$\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\theta_{t+1}^{(i)}) \approx \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\theta_{t+1}^{\text{SGD}})$$

- Less communications
- Cooperation is impossible if the drift is too large

One should not average gradients computed at different points

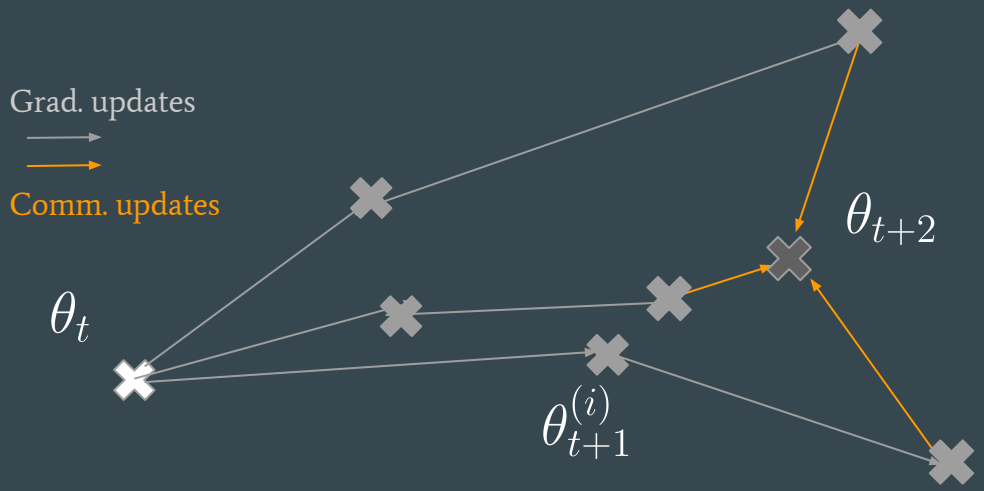


Agents 1 and 2 optimize the same objective

$$\nabla f(\theta_1) + \nabla f(\theta_2) = 0$$

Gradients can only be averaged if the parameters are not too far apart

“Federated Averaging”



Things that spread points apart:

- Functions heterogeneity
- Stochastic noise
- Learning rate
- Number of gradient steps

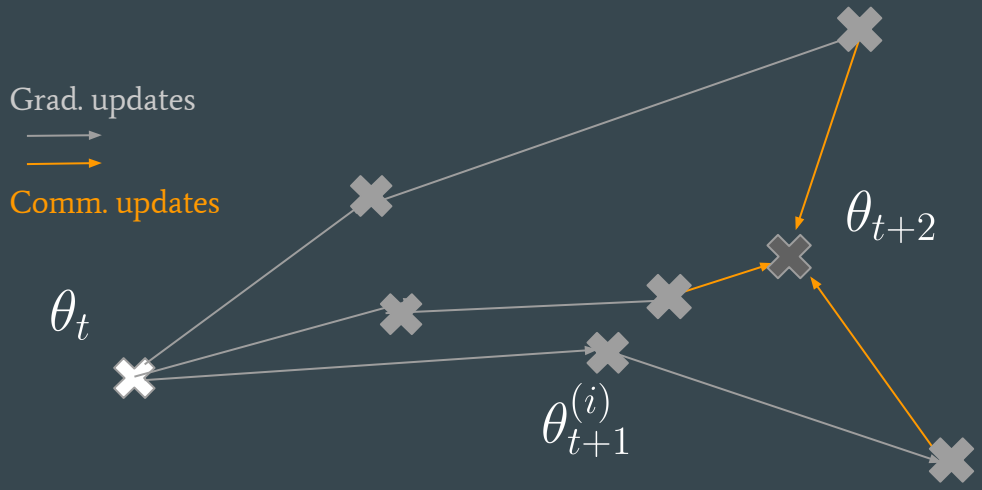
Things that make points closer:

- Communications

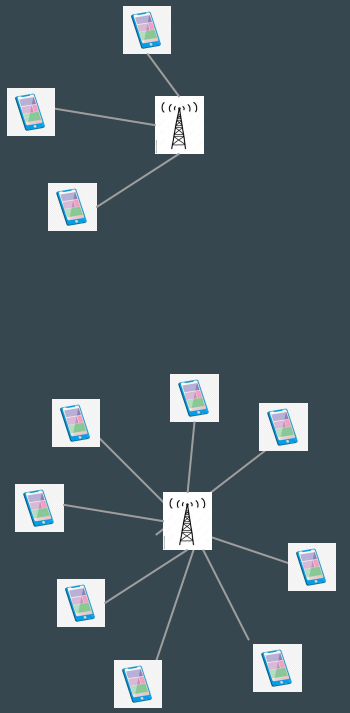
The game is to balance the number of communications to bound the drift

Several communication challenges

Scaling makes communications more expensive

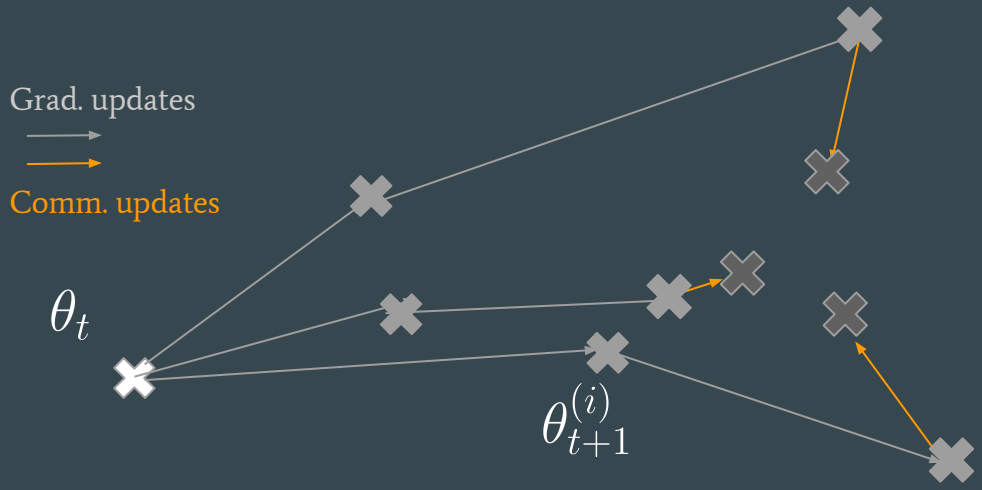


vs

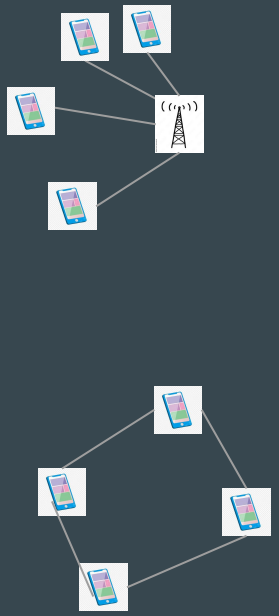


Communication bottleneck at the server

Decentralization complicates communications



VS



Local averages instead of global one.

Dropping global communications



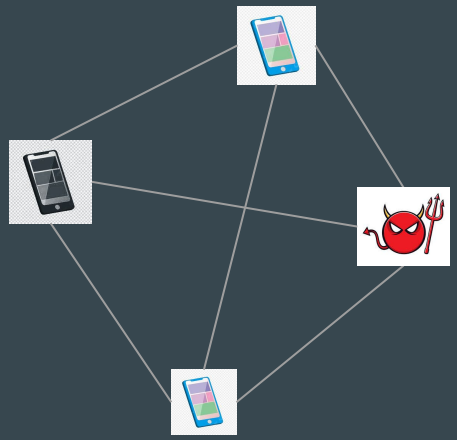
Partial averaging is performed instead of exact one.

Very costly to guarantee that everyone's parameter is close to the mean

Will be the focus of part I

What about faulty nodes?

Resilience to adversarial attacks



Byzantine agents: a fraction of the nodes can send arbitrary information.

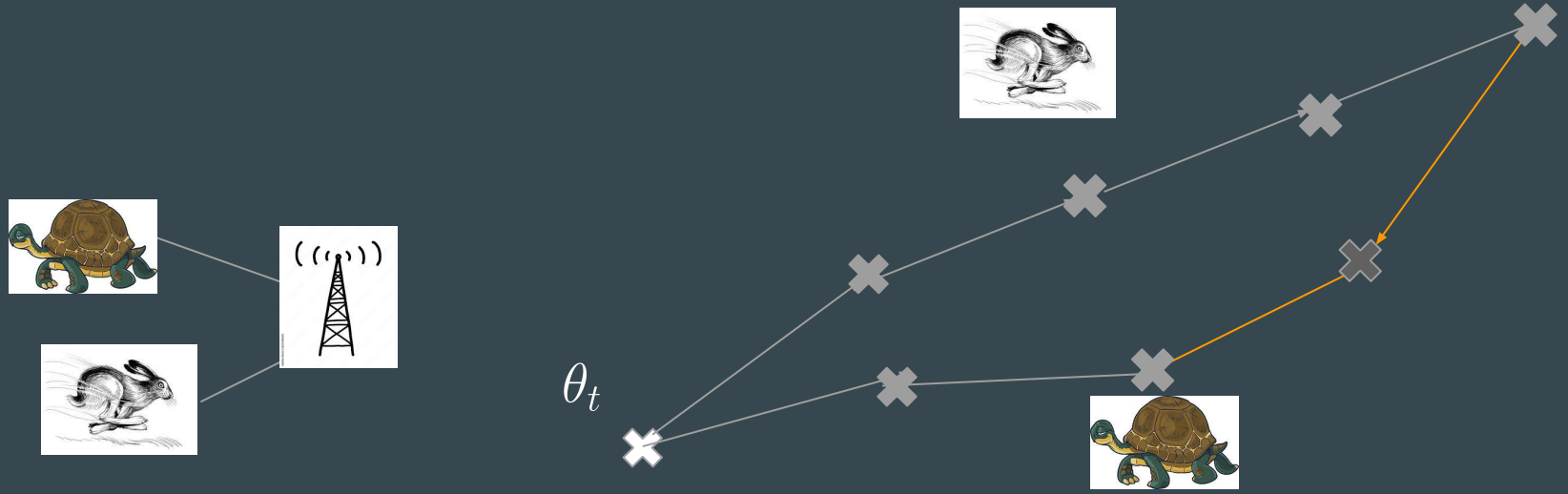
- computation with untrusted nodes

- arbitrary failures

Will be the focus of part II

How to deal with heterogeneity?

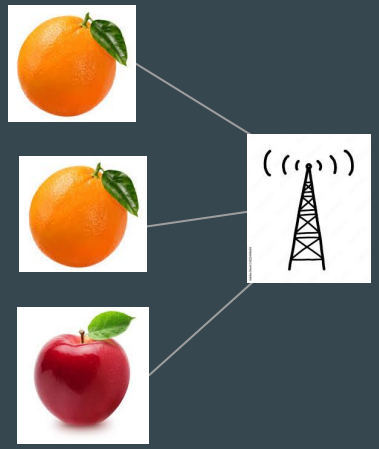
Heterogeneity: fast and slow nodes

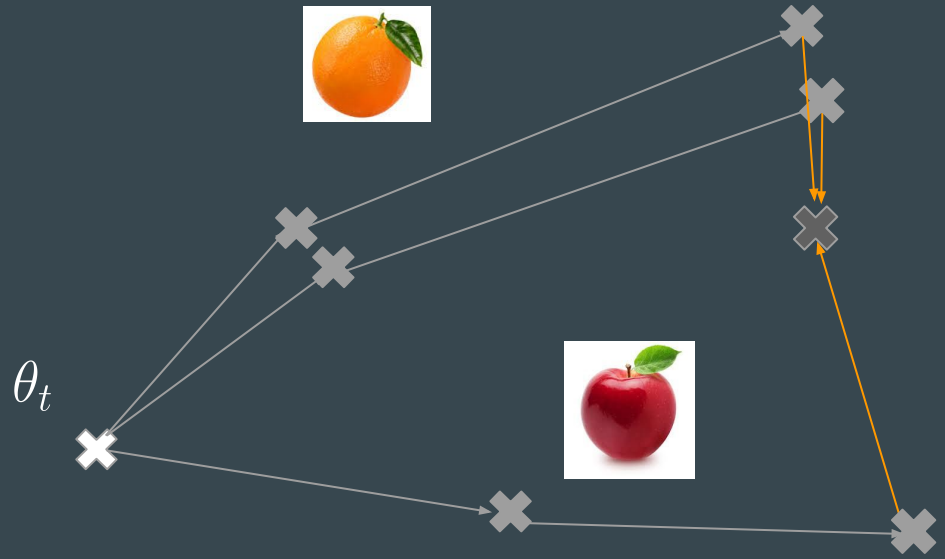


If the hare stops after 2 steps, computing power is wasted

If he continues, learning is biased towards the hare's data

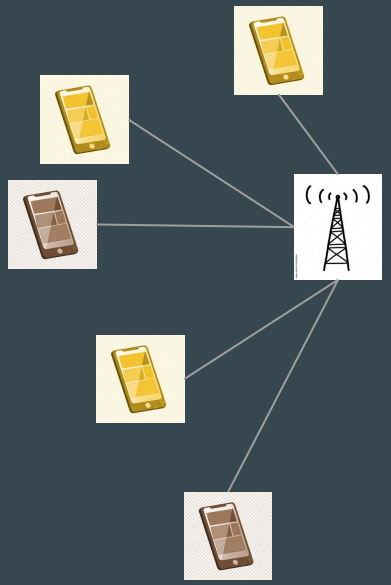
Heterogeneity: aggregating apples and oranges





Need very frequent communication to ensure that everyone is on the same page to be able to aggregate gradients

Personalization



Sometimes, the solution is to not give everyone the same model

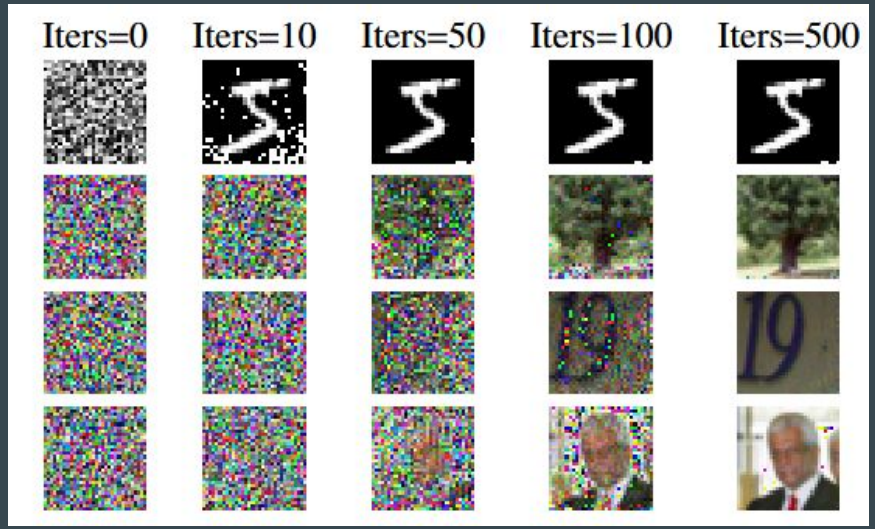
Plants identification depends on the region you are in.

How to decide who learns with whom without prior information?

Just because data doesn’t move doesn’t mean it’s not revealed

Privacy

Gradients potentially reveal as much information as actual data points.



Deep Leakage from Gradients, Zhu, Liu, Han, Neurips 2019

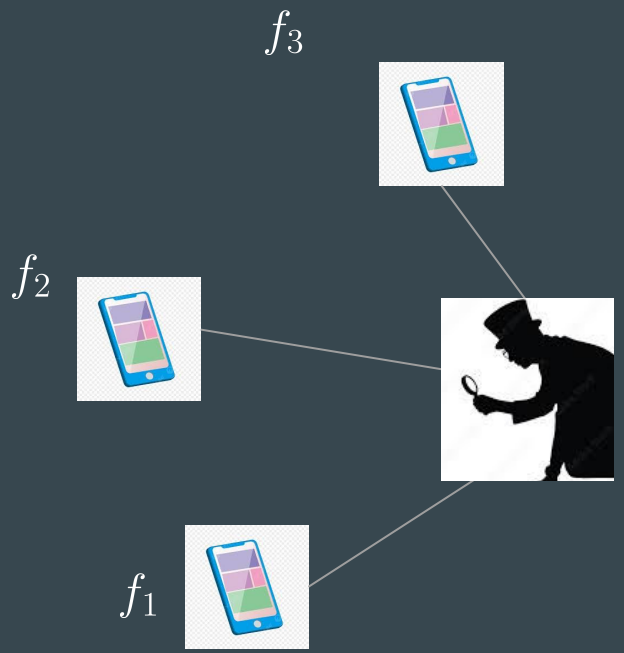
Privacy

One's data can be retrieved from the final model.

Example:

- You're the only billionaire in a village of about 1000 people.
- You participate in a study about how much money people have: people compute the mean of the inhabitant's net worth
- If the mean net worth is below 1 million, people will know you did not participate.
- If it is above, they will know you participated, and have a good estimate of how rich you are (if they manage to know how many people responded)

Privacy



Not sharing the data is not enough to protect privacy.

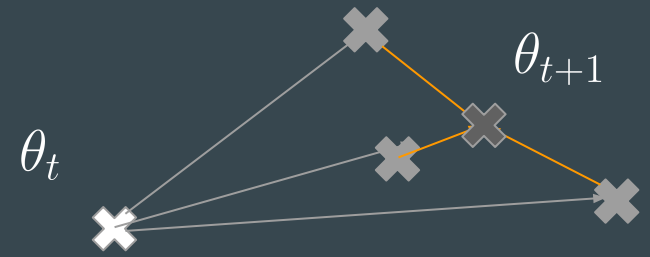
Solution: restrict the size of the gradients, and add noise proportional to this size.

=> Can bias learning substantially

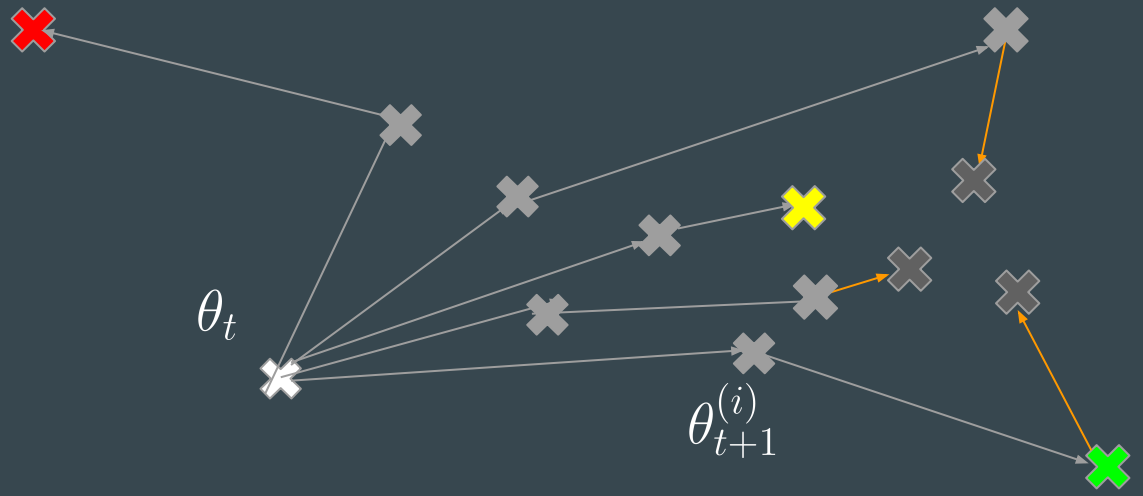
=> Very sensitive to the kind of noise that is added

Summing up

How you'd like it to look in practice



How it generally looks in practice



- Robustness
- Partial participation
- Heterogeneity
- Compression + Privacy
- Inexact averaging

The game is always to ensure that nodes' parameters remain somewhat close at the minimal possible cost, so that their information can be aggregated.

Plan of the talk

I - An introduction to distributed Optimization

II - Decentralized learning

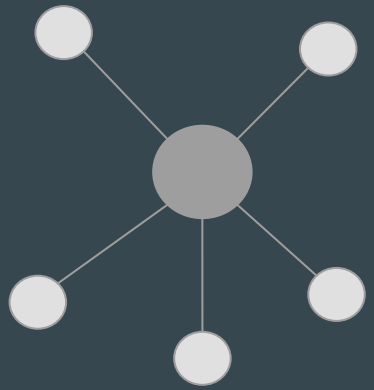
III - The role of topology in decentralized SGD

IV - Byzantine robustness in decentralized settings

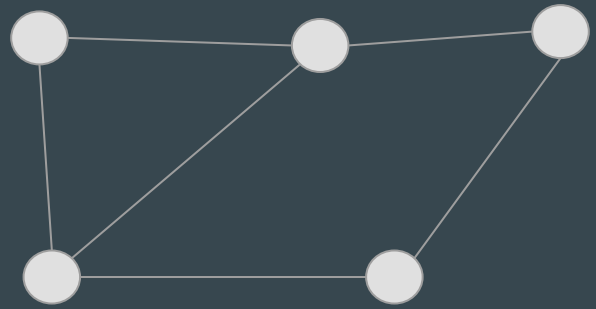
II - Decentralized learning

...

Towards decentralized architectures



- Single point of failure
- Communication bottlenecks



- Increased robustness
- Sparse lightweight communications
- Can favor privacy

Why consider the decentralized setting?

- It's more general

- **Often a good way to look at things if nodes are allowed to drift**
    - partial participation = decentralized on complete subgraphs
    - FedAvg = full communication every T computation steps.

- A promising way to tackle communication bottlenecks

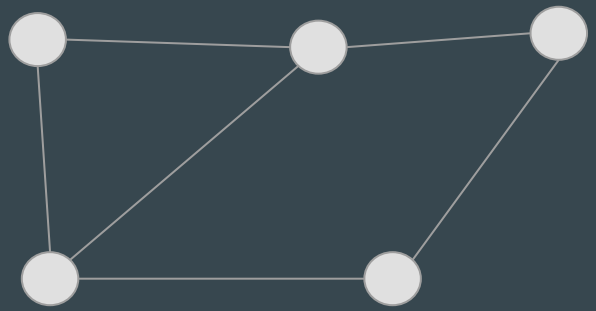
- Challenging algorithm design

Centralized Gradient Descent

$$\theta_{t+1} = \frac{1}{n} \sum_{i=1}^{n} (\theta_t - \eta \nabla f_i(\theta_t))$$

Update locally with (stochastic) gradients: requires compute
then average globally: requires communication

Decentralized communication model



- Nodes linked by a communication graph
- Local communications only

Computation step:

$$x_{t+1/2}^{(j)} = x_t^{(j)} - \eta \nabla f_{\xi_t^{(j)}}(x_t^{(j)})$$

Communication step:

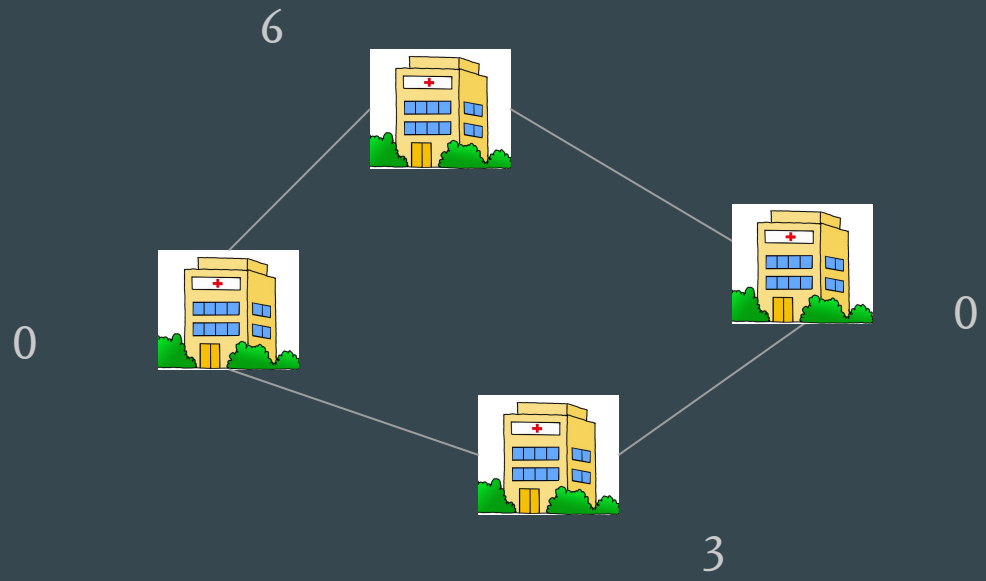
Gossip matrix

$$x_{t+1}^{(j)} = \sum_{i=1}^{n} W_{ij} x_t^{(i)}$$

Averaging is now **inexact**

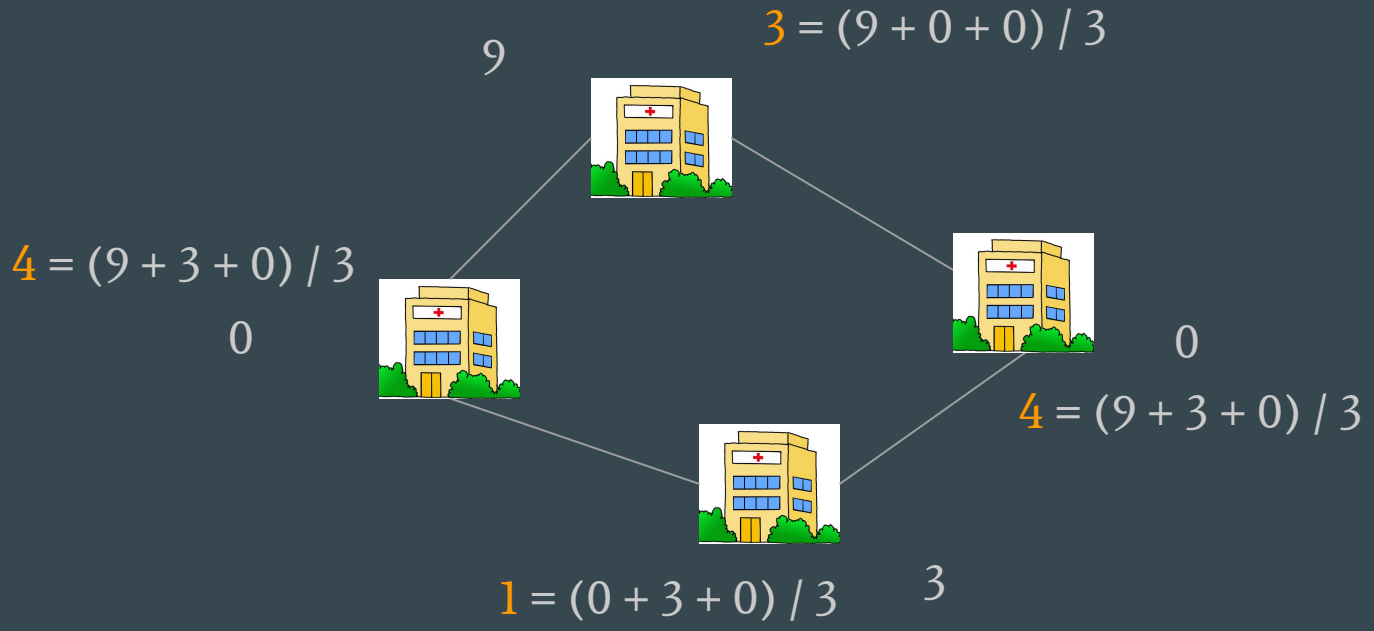
Gossip averaging

Basic version: nodes average with their neighbours.



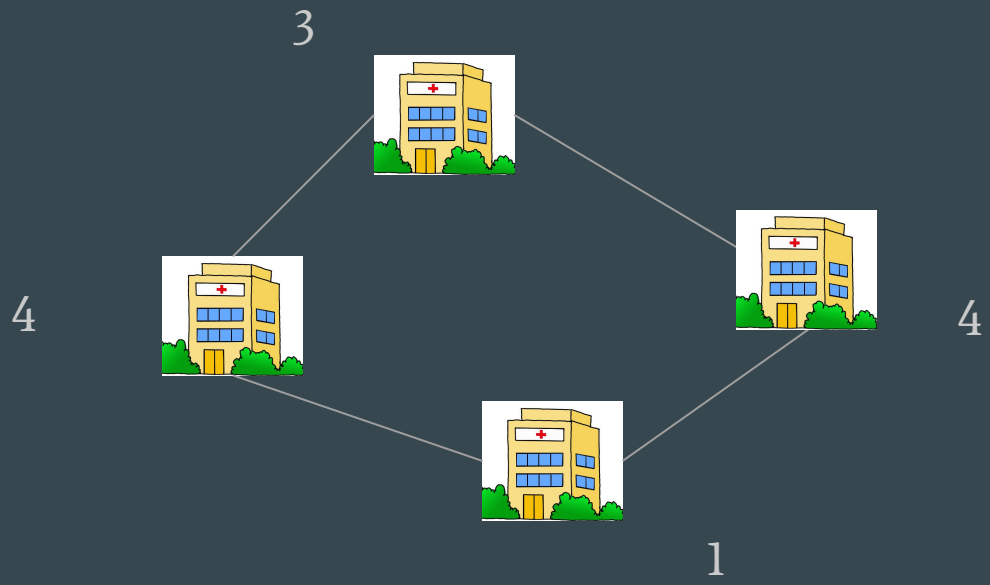
Gossip averaging

Basic version: nodes average with their neighbours.



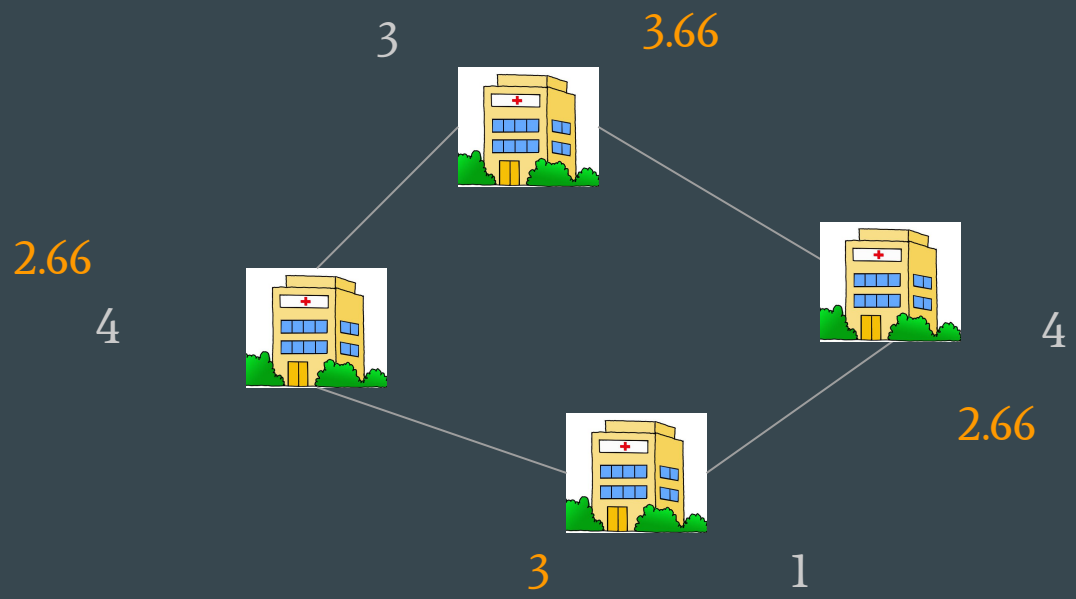
Gossip averaging

Basic version: nodes average with their neighbours.



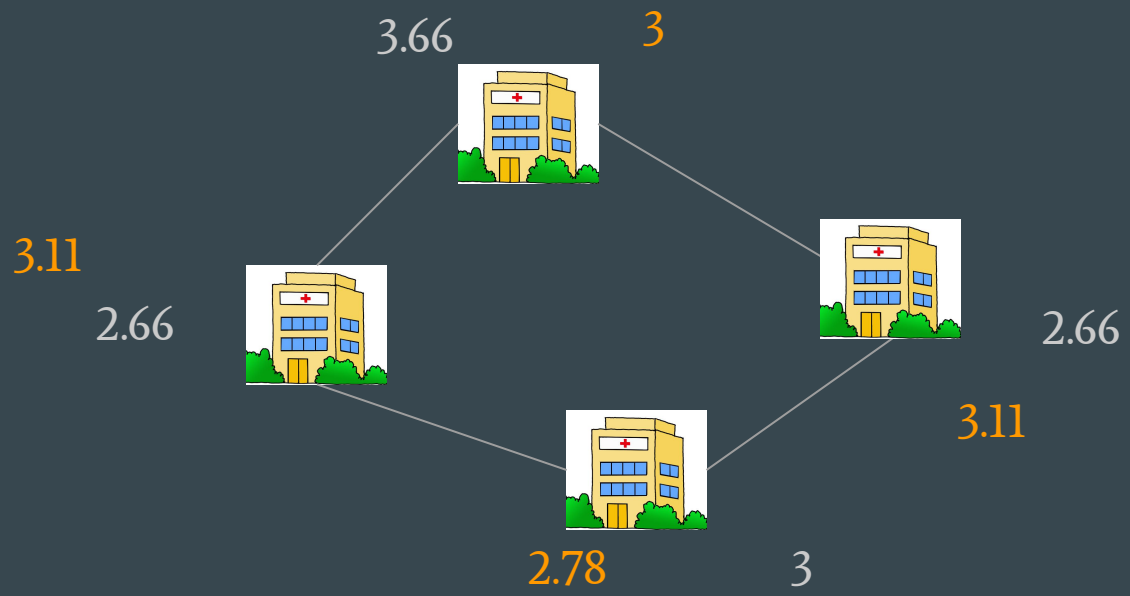
Gossip averaging

Basic version: nodes average with their neighbours.

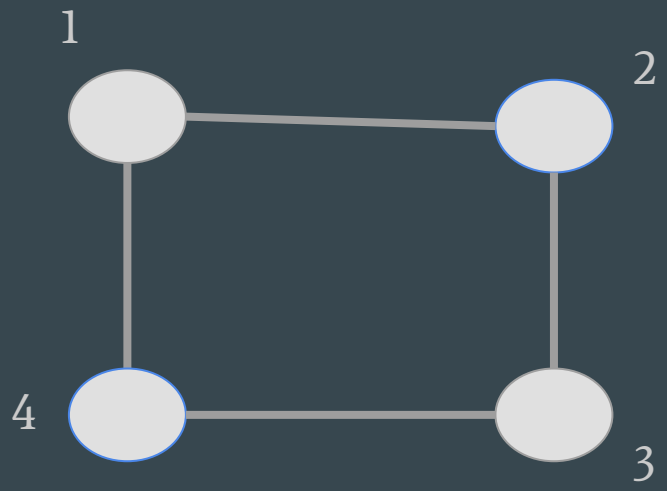


Gossip averaging

Basic version: nodes average with their neighbours.



The gossip matrix



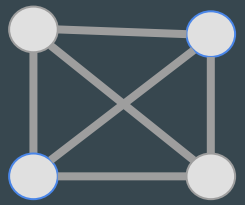
$$\frac{1}{3} \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

- Same sparsity pattern as the graph
- Row + Column stochastic

$$W1 = 1$$

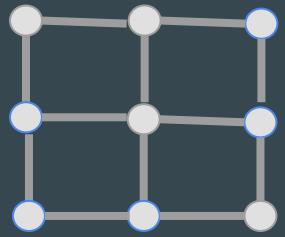
How good is W? Introducing the spectral gap

Averaging with W takes time $O((1-\lambda_2)^{-1})$



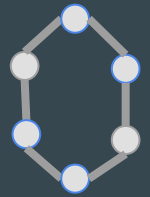
Fully connected

$1-\lambda_2 = O(1)$



Grid

$1-\lambda_2 = O(n^{-1})$



Ring

$1-\lambda_2 = O(n^{-2})$

$\lambda_2 \to 1$ as the size of the graph grows

Questions?

III - The role of topology in decentralized learning

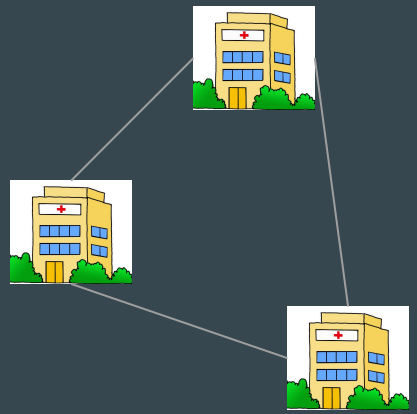
•••

Based on joint work with Thijs Vogels and Martin Jaggi





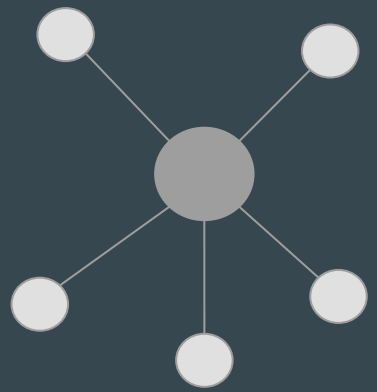
Distributed Optimization for Machine Learning



$$\min_{x \in \mathbb{R}^d} f(x)$$

At time t, node j computes a stochastic gradient $\nabla f_{\xi_t^{(j)}}(x_t^{(j)})$

Centralized communication model



- All nodes connected to a central server
- Can emulate single-machine algorithms

Computation step:

Learning rate

Stochastic gradient

$$x_{t+1/2}^{(j)} = x_t^{(j)} - \eta \nabla f_{\xi_t^{(j)}}(x_t^{(j)})$$

Communication step:

Number of nodes

$$x_{t+1}^{(j)} = \frac{1}{n} \sum_{i=1}^{n} x_{t+1/2}^{(i)}$$

Equivalent to mini-batch SGD!

A step back: minimizing an isotropic Gaussian

Goal: solve $\min_{x \in \mathbb{R}^d} \frac{1}{2}\|x\|^2$ $\|x_t - x_\star\|^2 \leq (1-\eta)^t$

GD: $x_{t+1} = (1-\eta)x_t$ $\eta \leq 1$

A step back: minimizing an isotropic Gaussian

Goal: solve $\min_{x \in \mathbb{R}^d} \frac{1}{2}\|x\|^2$ $\|x_t - x_\star\|^2 \leq (1-\eta)^t$

GD: $x_{t+1} = (1-\eta)x_t$ $\eta \leq 1$

SGD: $x_{t+1} = x_t - \eta\xi\xi^\top x_t$ $\eta \leq (d+2)^{-1}$ $\xi \sim \mathcal{N}(0, I_d)$

A step back: minimizing an isotropic Gaussian

Goal: solve $\min_{x \in \mathbb{R}^d} \frac{1}{2}\|x\|^2$ $\|x_t - x_\star\|^2 \leq (1-\eta)^t$

GD: $x_{t+1} = (1-\eta)x_t$ $\eta \leq 1$

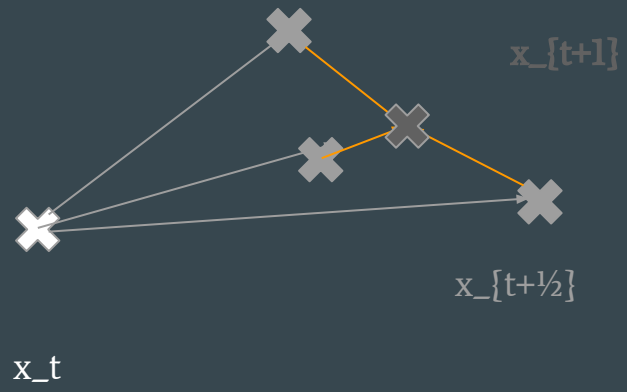
SGD: $x_{t+1} = x_t - \eta\xi\xi^\top x_t$ $\eta \leq (d+2)^{-1}$ $\xi \sim \mathcal{N}(0, I_d)$

Minibatch: $x_{t+1} = x_t - \frac{\eta}{n}\sum_{i=1}^{n} \xi_i\xi_i^\top x_t$ $\eta \leq \frac{n}{n+d+1}$

Collaboration → Large learning rate

- Faster convergence
- Better generalization?

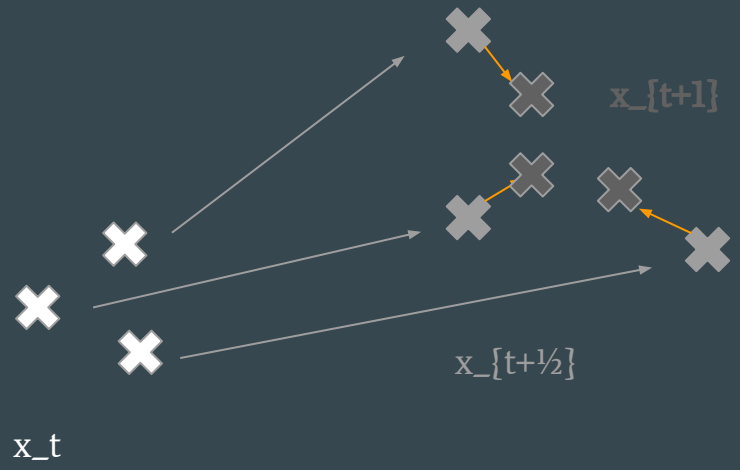
Learning rate tradeoffs



Gradient step increases the distance between points and their average

Averaging step brings points back together

Learning rate tradeoffs



Gradient step increases the distance between points and their average

Gossip step **approximately** brings points back together

Large learning rates lead to more dispersion

Problem: variance cannot be reduced if stochastic gradients are computed at (very) different points.

“Classical” decentralized theory, Koloskova et. al. (2020)



“Classical” theory, Koloskova et. al. (2020)



- Alone we go faster

Variance

Condition number

Bias

$$\frac{\sigma^2}{\mu\varepsilon} + \kappa \log \varepsilon^{-1}$$

Strong convexity

Final accuracy

“Classical” decentralized theory, Koloskova et. al. (2020)



- Alone we go faster

Variance

Condition number

Bias

Strong convexity

Final accuracy

$$\frac{\sigma^2}{\mu\varepsilon} + \kappa \log \varepsilon^{-1}$$

- Together, we go further

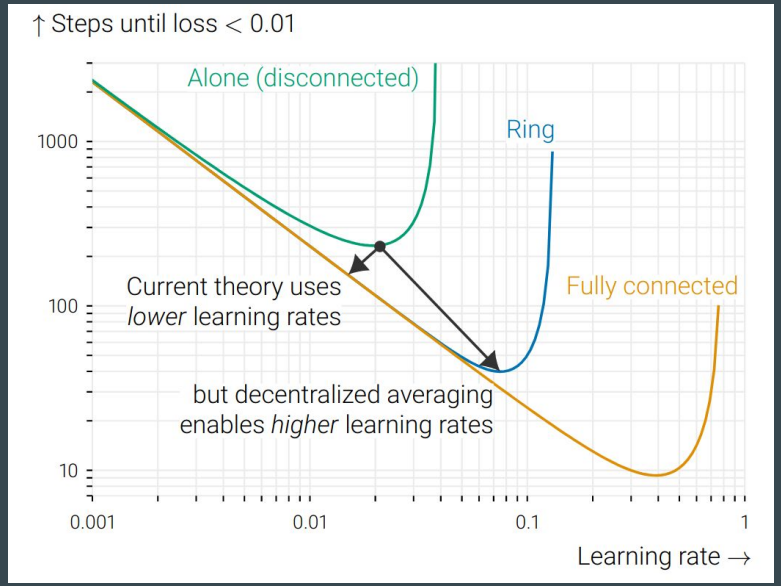
$$\frac{\sigma^2}{\mu\varepsilon n} + \frac{\kappa}{1-\lambda_2} \log \varepsilon^{-1}$$

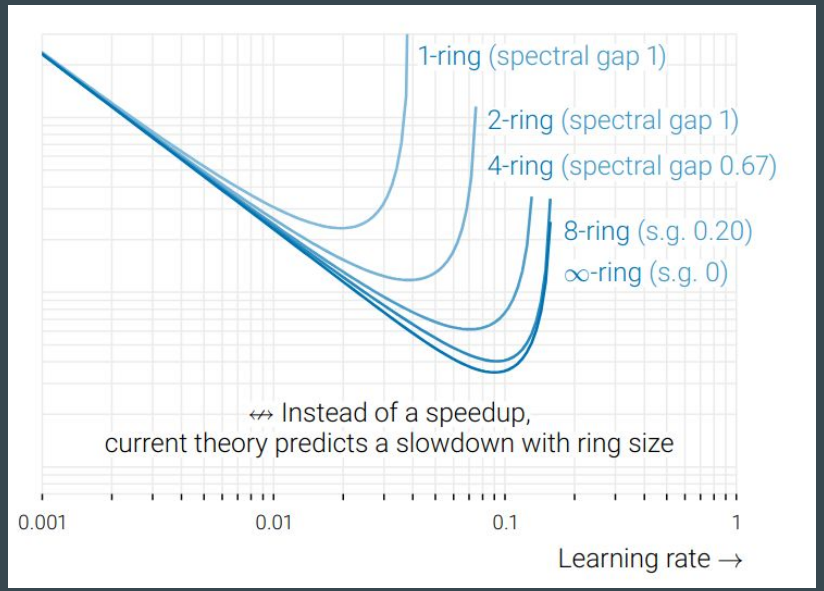
The learning rate is reduced in order to enable collaboration

Collaboration → Large learning rate

??????

Collaboration → large learning rate





Our goal is to come up with a new theory that better describes this behaviour

How good is W at averaging??

Random walk: $y_{t+1} = \sqrt{\gamma} \cdot y_t + \xi_t$

$\xi_t \sim \mathcal{N}(0, I_n)$

Averaged version: $z_{t+1} = W(\sqrt{\gamma} \cdot z_t + \xi_t)$

$$n_W(\gamma) = \lim_{t \to \infty} \frac{\text{var } y_t}{\text{var } z_t}$$

How much does averaging by W reduce the variance of the random walk process?

The effective number of neighbors

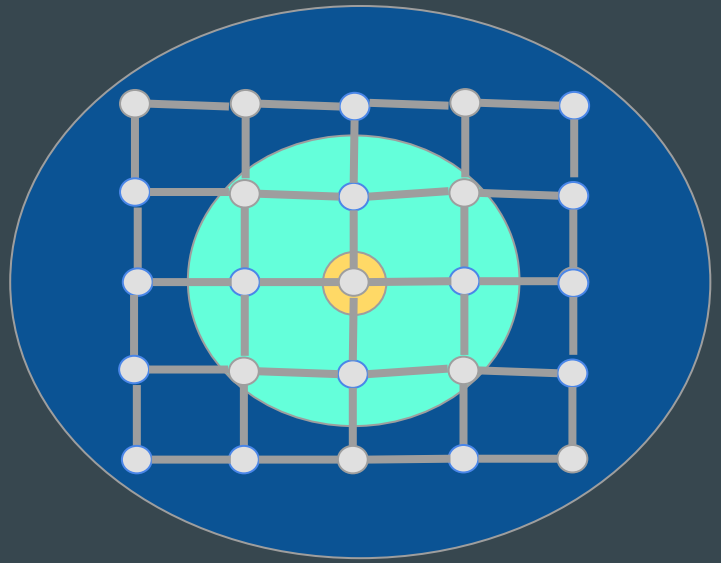
$$1 \leq n_W(\gamma) \leq n$$

$$n_W(\gamma) = \frac{\frac{1}{1-\gamma}}{\frac{1}{n}\sum_{i=1}^{n} \frac{\lambda_i^2}{1-\lambda_i^2\gamma}}$$

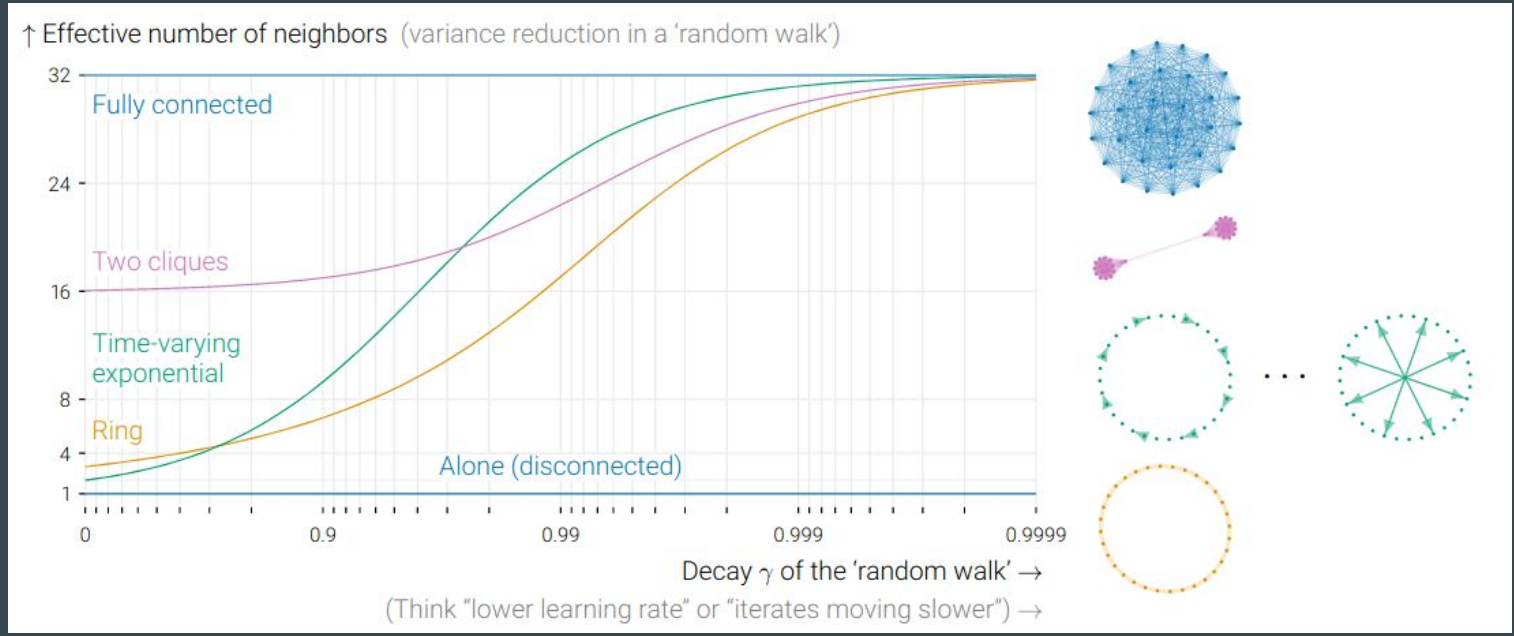
$$M = (1-\gamma)\sum_{k=1}^{\infty} \gamma^{k-1} W^{2k}$$

$$M_{ii} = n_W(\gamma)^{-1}$$

Small gamma Medium gamma Large gamma



The effective number of neighbors



Links with spectral dimension

Spectral dimension d_s of a graph: how fast do eigenvalues decay?

Coincides with “expected” dimension (i.e.: line = 1, grid = 2, cube = 3...).

For $d_s > 2$:

$$n_W(\gamma) \geq \left(\frac{1}{n} + \frac{1-\gamma}{\gamma(d_s - 2)c_s} \right)^{-1}$$

The expected number of neighbours scales linearly with the spectral dimension of the graph (up to 1/n).

The two variances

!! Homogeneity: expected function is the same for all nodes !!

- Stochastic gradients $\nabla f_{\xi_t^{(j)}}$, with $\mathbb{E}\left[f_{\xi_t^{(j)}}\right] = f$ for all t, j.
- "Standard" variance $\sigma^2 = \|\nabla f_\xi(x_\star)\|^2$ ← How different are the optima??
- Each $f_{\xi_t^{(j)}}$ is ζ smooth, f is L-smooth ← How regular are the stochastic gradients compared to the global one?

The two variances

!! Homogeneity: expected function is the same for all nodes !!

- Stochastic gradients $\nabla f_{\xi_t^{(j)}}$, with $\mathbb{E}\left[f_{\xi_t^{(j)}}\right] = f$ for all t, j.
- "Standard" variance $\sigma^2 = \|\nabla f_\xi(x_\star)\|^2$ ← How different are the optima??
- Each $f_{\xi_t^{(j)}}$ is ζ smooth, f is L-smooth ← How regular are the stochastic gradients compared to the global one?

$$\zeta \gg L$$

Ex: $\nabla f_\xi(x_t) = \xi\xi^\top x_t$, with $\xi \sim \mathcal{N}(0, I_d)$,

then $\zeta = d + 2$ and $L = 1$

ζ is the amount of noise (far from optimum) in the stochastic gradients

Convergence Results

Step-size constrained by noise

Step-size constrained by (approximate) consensus constraint

$$\eta \leq \min\left(\frac{1}{\frac{\zeta}{n_W(\gamma)} + L}, \frac{1-\gamma\lambda_2}{n_W(\gamma)L}\right)$$

$$\|x_t - x_\star\|_M^2 = O\left((1-\eta\mu)^t + \frac{\eta\sigma^2}{n_W(\gamma)}\right)$$

Convergence Results

Step-size constrained by noise

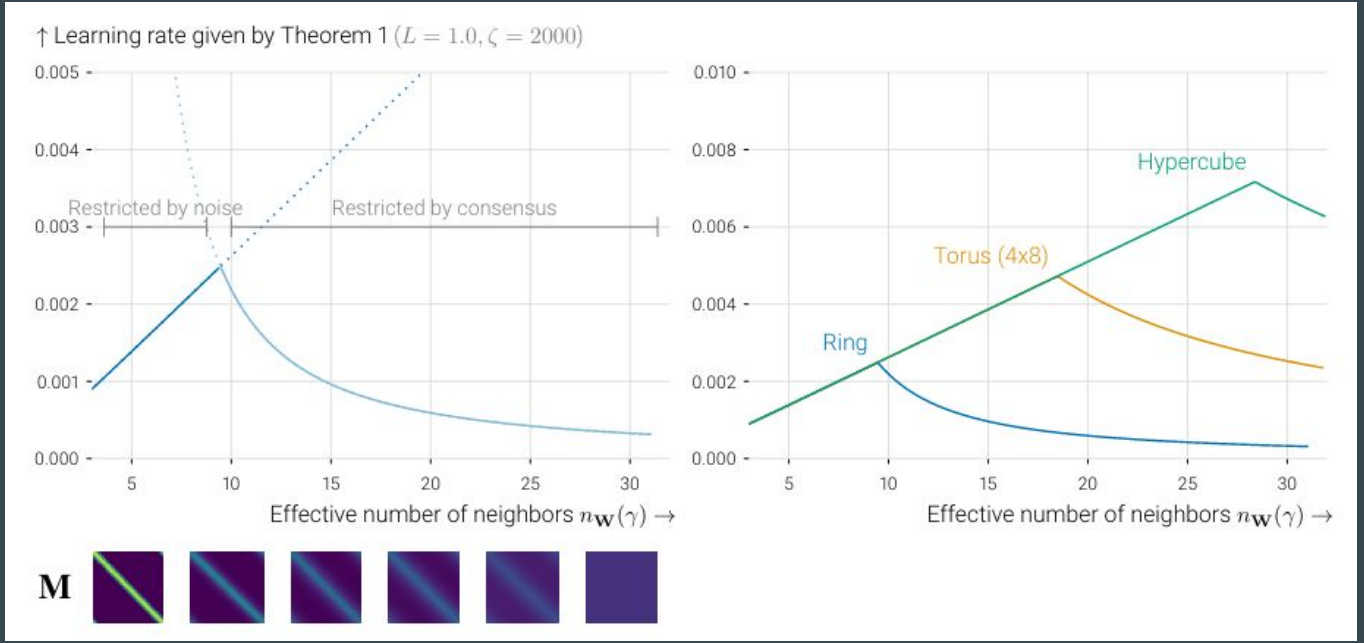
Step-size constrained by (approximate) consensus constraint

$$\eta \leq \min\left(\frac{1}{\frac{\zeta}{n_W(\gamma)} + L}, \frac{1-\gamma\lambda_2}{n_W(\gamma)L}\right)$$

$$\|x_t - x_\star\|_M^2 = O\left((1-\eta\mu)^t + \frac{\eta\sigma^2}{n_W(\gamma)}\right)$$

- D-SGD has the same rates as (centralized) minibatch SGD with $n_W(\gamma)$ nodes
- Communication and larger graphs always help
- Eventual dependence on the spectral gap to get $n_W(\gamma) \approx n$

How to tune the effective number of neighbours



In practice, you don't have to know γ and just tune the step-size

Reducing the step-size reduces the variance in two ways

Variance: $\dfrac{\eta\sigma^2}{n_W(\gamma)}$

- This term shrinks when we reduce η
- The analysis should consider the largest γ such that $\eta \leq \dfrac{1-\gamma\lambda_2}{n_W(\gamma)L}$

  so decreasing η allows to increase γ, which also increases $n_W(\gamma)$

The variance term decreases super-linearly with the step-size

Other insights we can derive :
Either increase the connectivity or the number of nodes

- $n_W(\gamma) < n$: Bigger graphs can only help...
  ... but it's useless to increase the size of the graph if learning rates are too large

  => Nodes effectively only "see" a local neighborhood

  => Increase connectivity

- $n_W(\gamma) \approx n$: Using more connected topologies is not always helpful

  => Depending on the level of noise in the problem, nodes might already "see" the full graph even with sparsely connected topologies

  => Increase the number of nodes / reduce the number of communication rounds

Idea of the proof

- Standard analyses look at $\left\|\bar{x}_t - x_\star\right\|^2$, thus requiring all nodes to be close.
- Instead, we look at $\left\|x_t - x_\star\right\|_M^2$, with $M = (1-\gamma)\sum_{k=1}^{\infty}\gamma^{k-1}W^{2k}$
- Variance: $\mathbb{E}\|\nabla f_\xi(x)\|_M^2 = \sum_i M_{ii}\mathbb{E}\|\nabla f_{\xi^{(i)}}(x^{(i)})\|^2 + \|\nabla f(x)\|_{M-D_M}^2$

Idea of the proof

$$\|x_{t+1} - x_\star\|_M^2 \leq (1 - \eta\mu)\|x_t - x_\star\|_M^2 + \eta^2\sigma_M^2$$
$$-2\eta(3/4 - \eta\left[\zeta/n_W(\gamma) + L\right])D_f(x_\star, x_t)$$
$$+\eta L\|Mx_t - x_t\|^2$$

Reduced variance term

Error term, step-size condition

Additional error term proportional to how far iterates are from their (locally averaged) mean

Idea of the proof

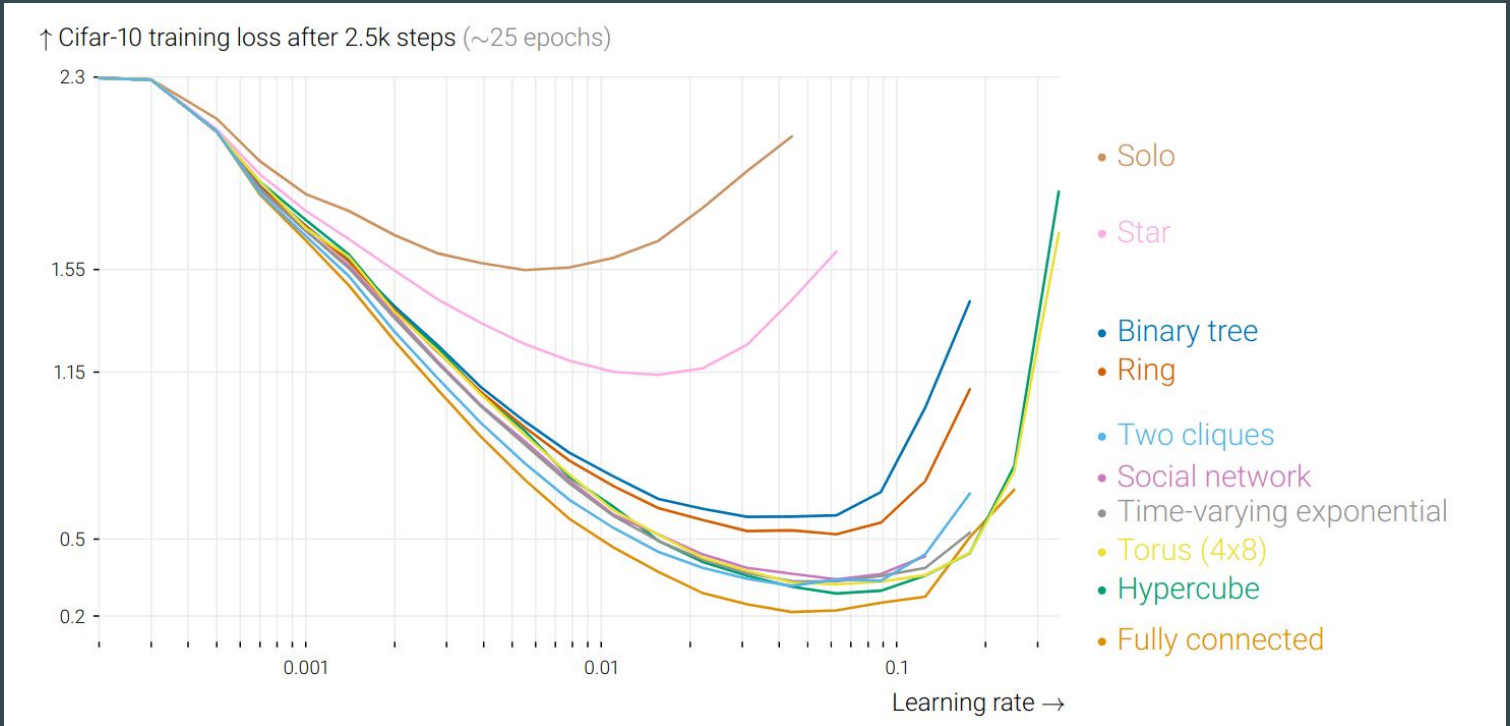
- We control the excess terms by looking at the evolution of $\|x_t\|^2_{I-M}$

- We have: $\|x_{t+1}\|^2_{I-M} \leq (1-\eta\mu)\|x_t\|^2_{I-M} + 2\eta\alpha D_f(x_\star, x_t)$
$$-\beta\|Mx_t - x_t\|^2$$

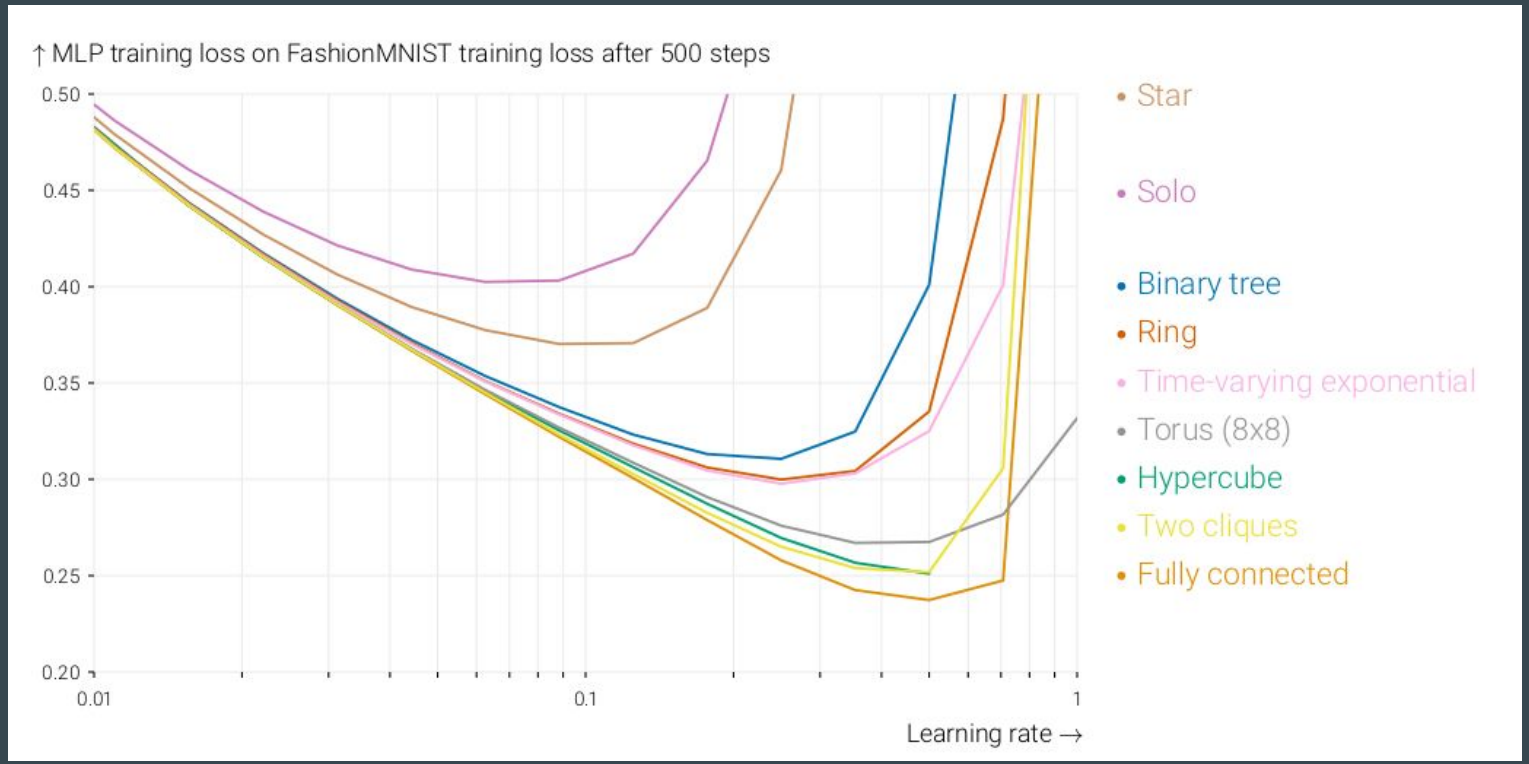
  where: $I - W \succcurlyeq \beta(I-M)$

How about in practice?

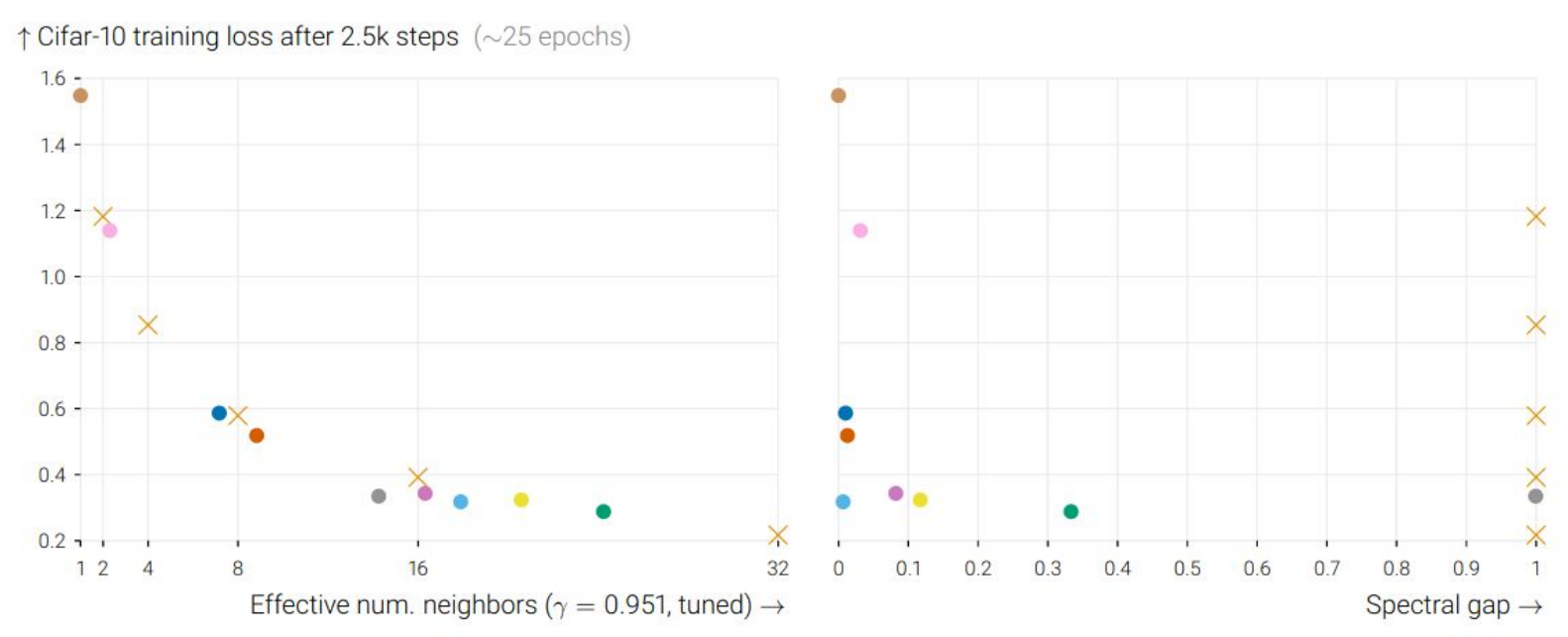
Deep Learning Experiments



Deep Learning Experiments - bis

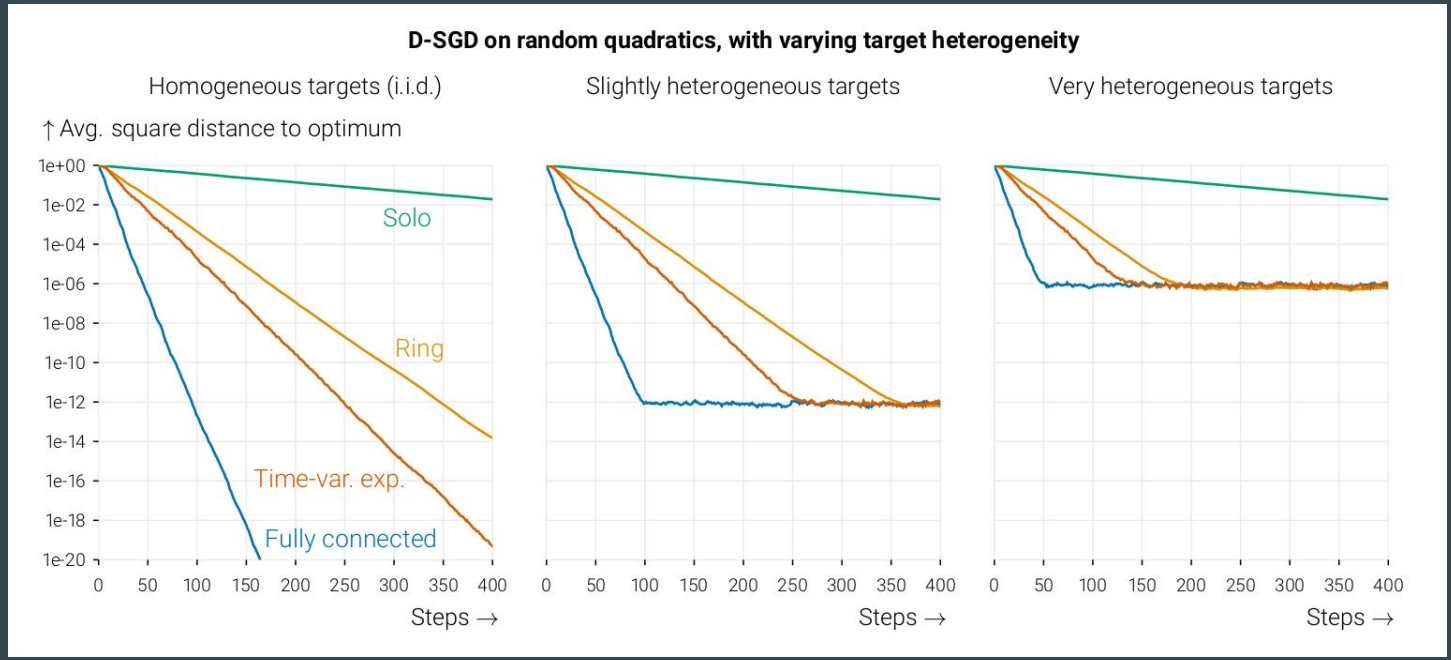


Comparison with spectral gap



The topology only affects performance through the number of effective neighbors

Breaking the homogeneity assumption



Heterogeneity only changes the behaviour up to a ball around the optimum

The same thing holds for heterogeneous setting!

$$\eta \leq \min\left(\frac{1}{\frac{\zeta}{n_W(\gamma)} + L}, \frac{1-\gamma\lambda_2}{n_W(\gamma)L}\right)$$

$$\|x_t - x_\star(\eta)\|_M^2 = O\left((1-\eta\mu)^t + \frac{\eta\sigma^2}{n_W(\gamma)}\right)$$

- The fixed point of the (mean) dynamics does not respect consensus anymore.
- Extra “variance” due to the local optima being different.
- Can be controlled as well, and depends on the interplay of the graph topology and the heterogeneity structure around the optimum.

Conclusion

- Communication allows for larger learning rates:
  => Reconciled theory and practice
- Connections between D-SGD and centralized mini-batch SGD
- The effective batch size depends on the topology and on the learning rate
- Theory is consistent with infinite graphs

Open directions

- Theory for non-convex (experiments suggest yes for deep learning)
- Better understanding the weakness of D-SGD for heterogeneous deep learning?
- Better understanding of the local learning dynamics.
  => Implicit bias of D-SGD?

Revisiting sayings





Together, we go both faster and further...

... and so does D-SGD

Questions?

IV - Unified breakdown analysis for Byzantine-robust gossip

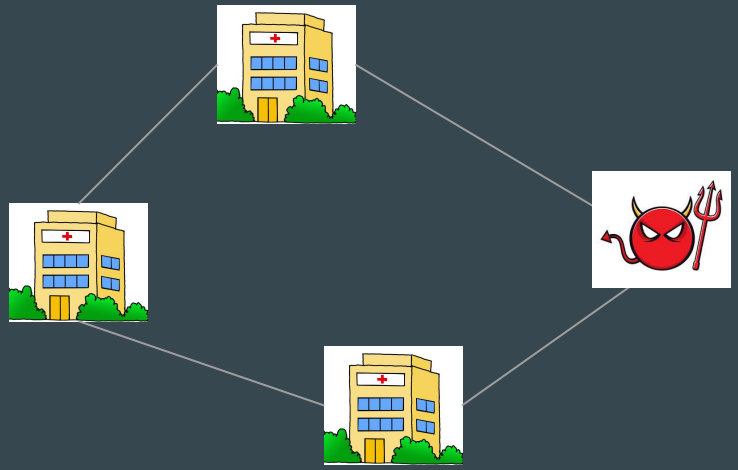
•••

Based on joint work with Renaud Gaucher and Aymeric Dieuleveut



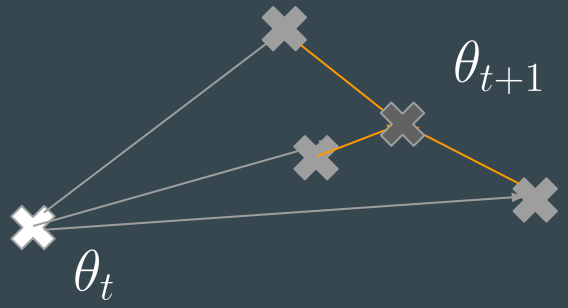


Byzantine-Robust Distributed Optimization (Lamport, 1982)



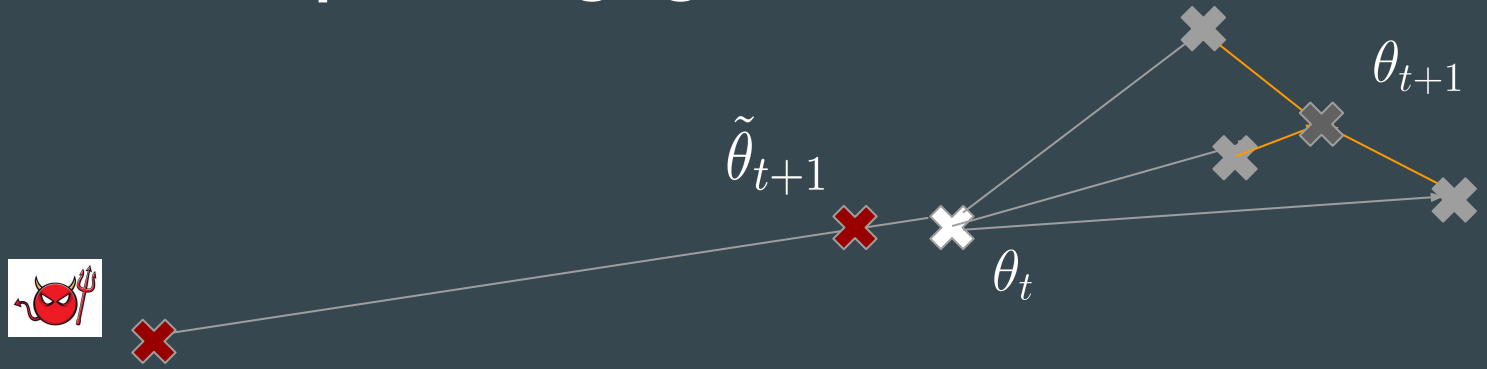
- Some adversaries disrupt learning.
  - may be on purpose
  - may be to model arbitrary faults
- Byzantine nodes can collude
- They have arbitrary computing power

Failure of simple averaging methods



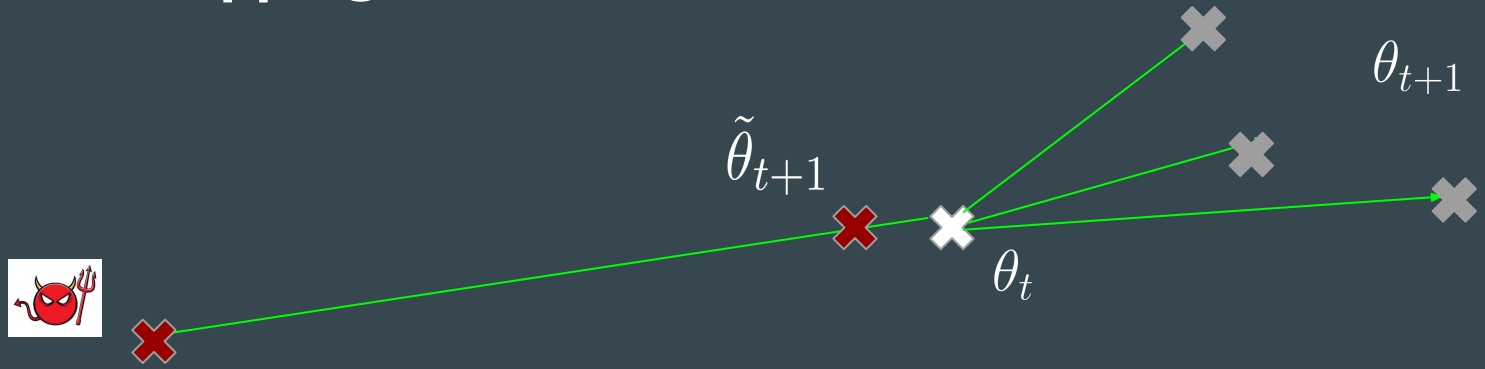
The mean is not robust: one node can arbitrarily perturb learning.

Failure of simple averaging methods

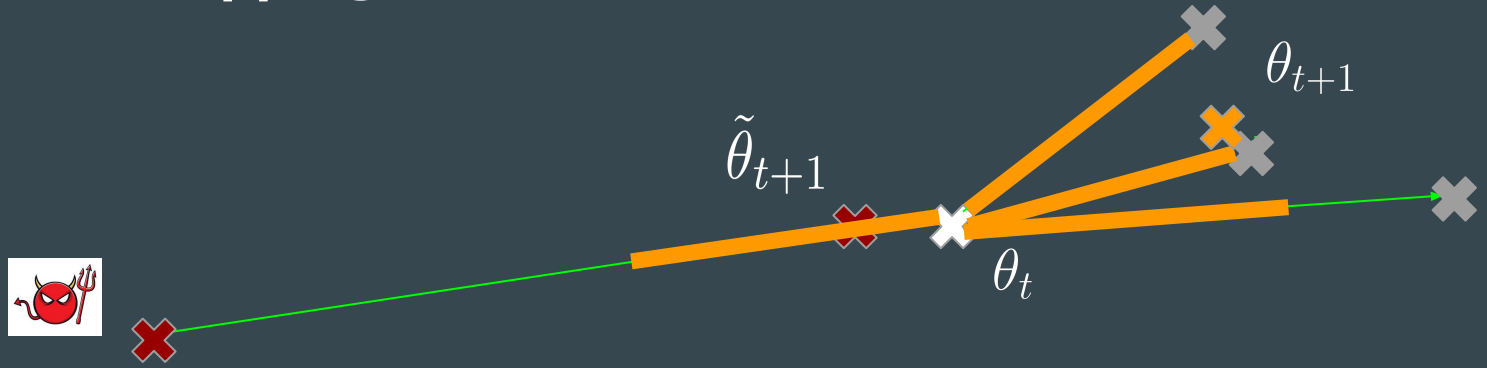


The mean is not robust: one node can arbitrarily perturb learning.

The fix: clipping contributions



The fix: clipping contributions



- Works if there are at least 2 / 3 of honest nodes
- Biases the learning process: impacts fairness
  (no way to distinguish malicious nodes from honest ones)

Other methods: compute the (geometric) median , dump the largest updates...

Why does it require honest nodes?

The clipping threshold is the key, and is very hard to set.

Constant threshold: errors accumulate, so the influence of Byzantine nodes depends on the number of epochs you make.

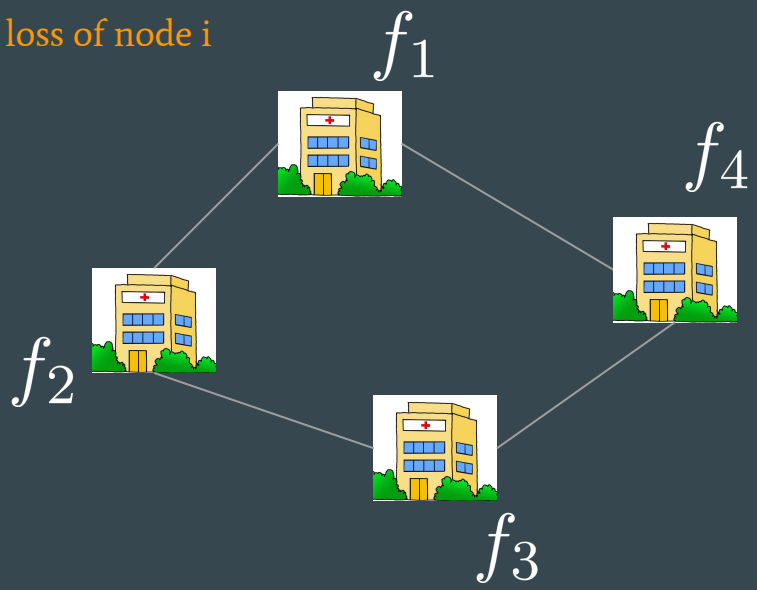
Diminishing: how to tune the decrease in order to stop at the right time (i.e., when close enough to the objective).

Distributed Optimization for Machine Learning

Number of nodes in the network

Local loss of node i

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

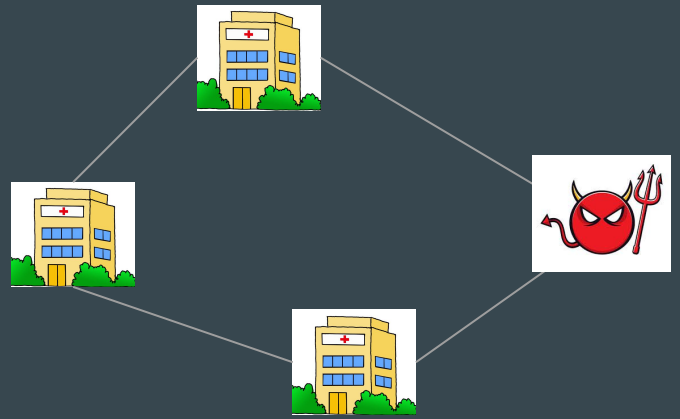


- Each node only has access to summands of the global objective function

Byzantine-Robust Distributed Optimization

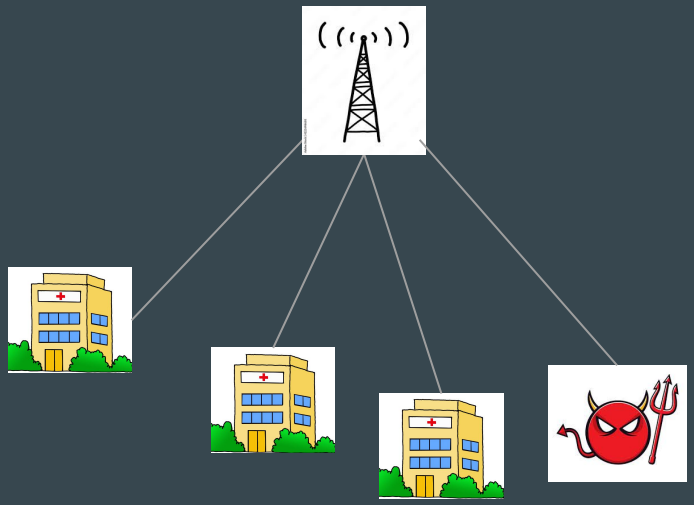
$$\min_{\theta \in \mathbb{R}^d} f(\theta) = \sum_{i \in H} f_i(\theta)$$

We are only interested in honest nodes



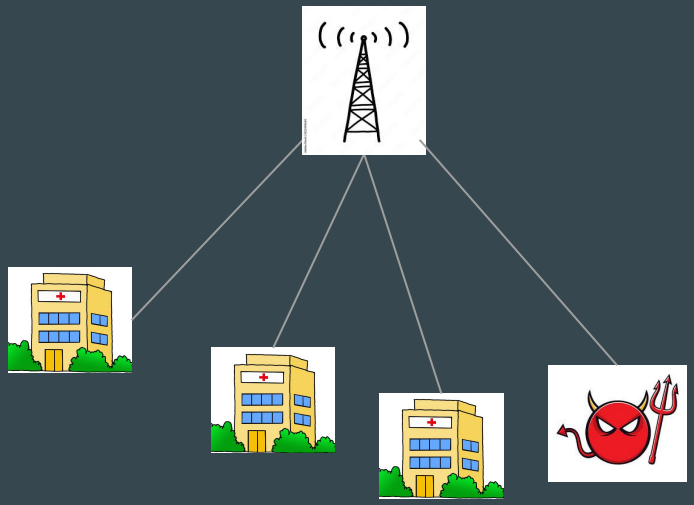
Problem: we don't know who's honest and who's not

Two communication models

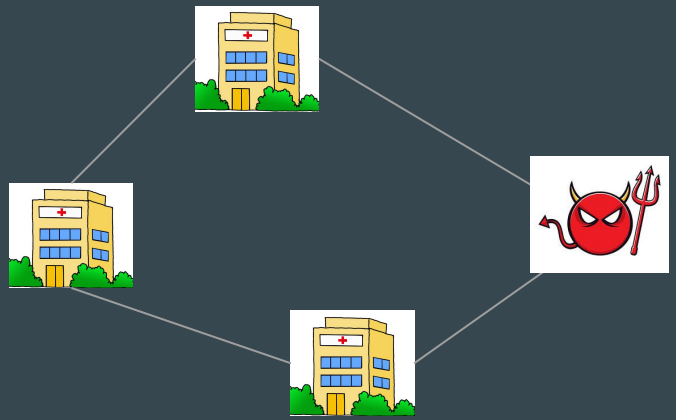


Centralized
(trusted central server)

Two communication models

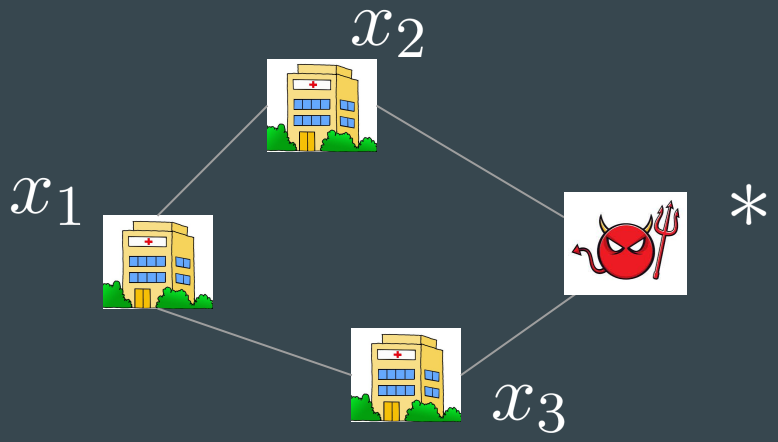


Centralized
(trusted central server)



Decentralized
(p2p communications)

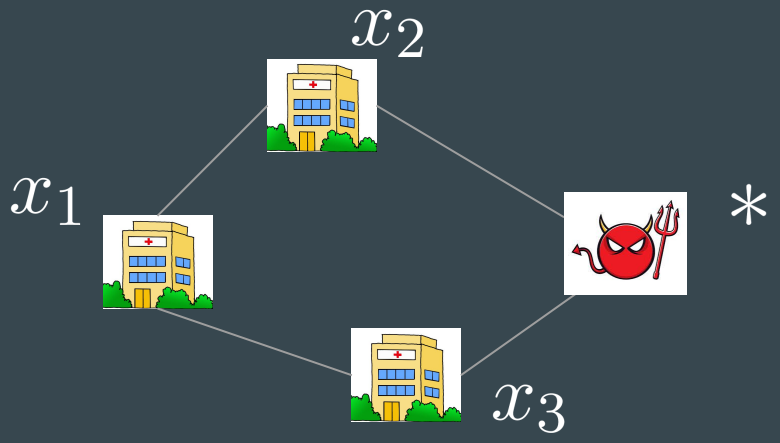
Subproblem: robust distributed averaging



Goal: find $\frac{1}{n_h} \sum_{i \text{ honest}} x_i$

$$\arg\min_{\theta \in \mathbb{R}^d} \sum_{i \text{ honest}} \|\theta - x_i\|^2$$

Subproblem: robust distributed averaging



Goal: find $\frac{1}{n_h} \sum_{i \text{ honest}} x_i$

$$\arg\min_{\theta \in \mathbb{R}^d} \sum_{i \text{ honest}} \|\theta - x_i\|^2$$

3 Reasons to like decentralized averaging

- Trade-off bias (error introduced by Byzantine nodes) for variance
- Retains all the complexity of the initial problem
- Reductions exist to solve the general problem from this

Robustness criterion

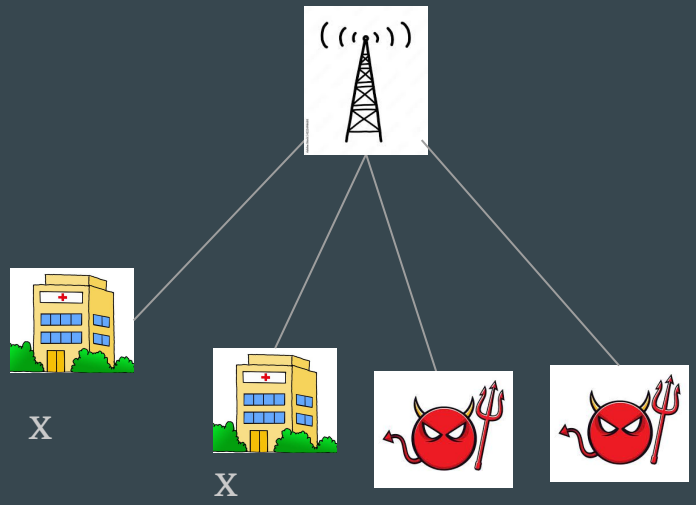
r-robustness:

$$\frac{1}{|H|}\sum_{i\in H}\|x_i^+ - \overline{x}_H\|^2 \leq r\frac{1}{|H|}\sum_{i\in H}\|x_i - \overline{x}_H\|^2.$$

The quadratic error should strictly (r<1) decrease

Maximal proportion of adversaries

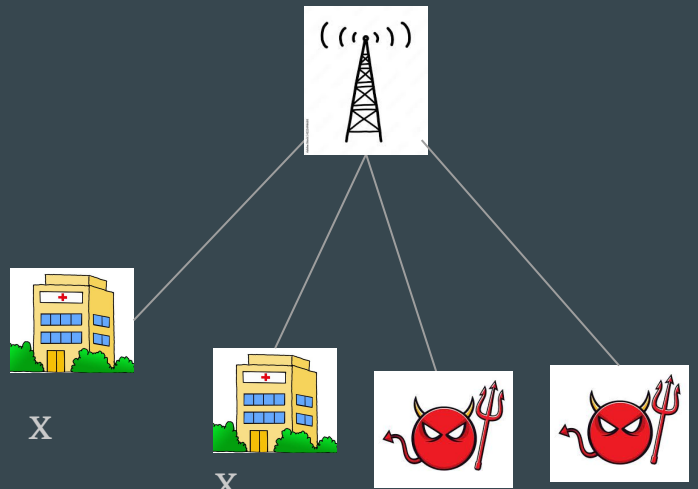
Federated (centralized): no robustness is possible if half or more byzantine nodes



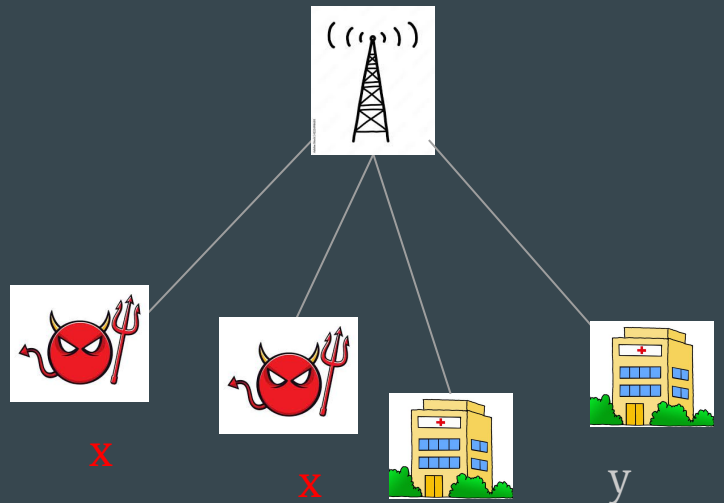
Setting 1

Maximal proportion of adversaries

Federated (centralized): no robustness is possible if half or more byzantine nodes



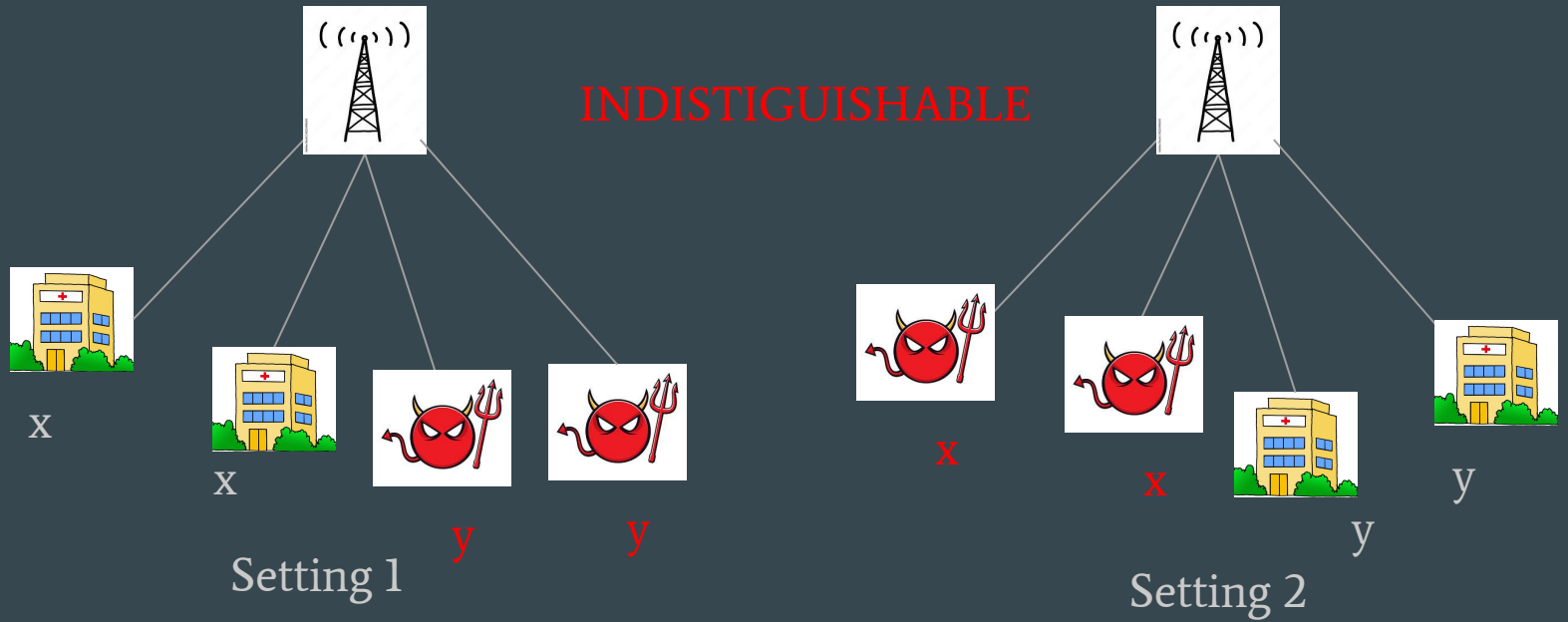
Setting 1



Setting 2

Maximal proportion of adversaries

Federated (centralized): no robustness is possible if half or more byzantine nodes

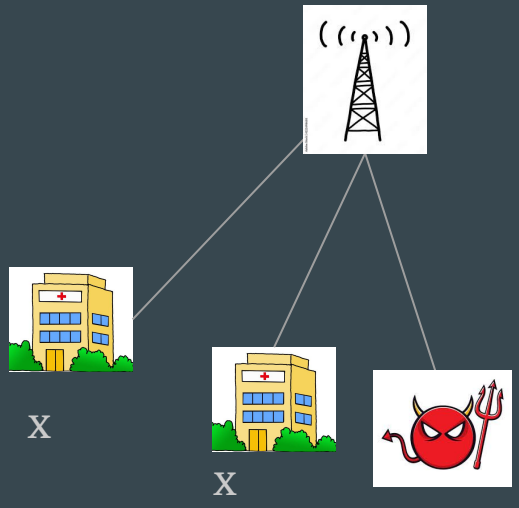


Minimal error

The error depends on the heterogeneity between the honest nodes

Minimal error

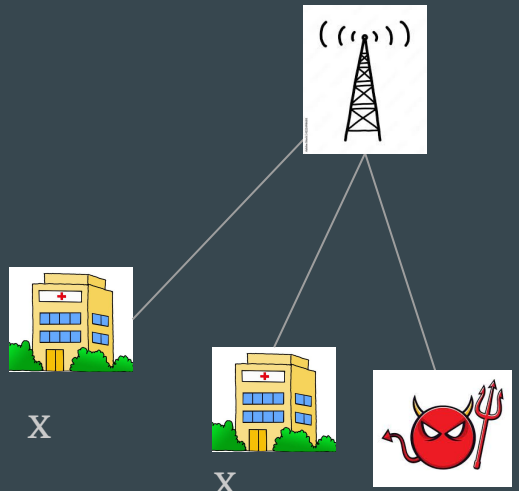
The error depends on the heterogeneity between the honest nodes



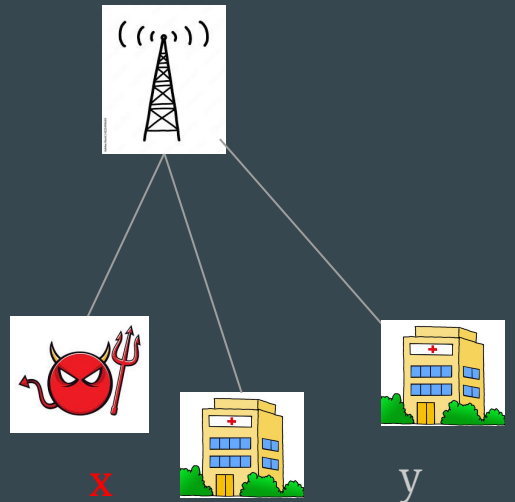
Setting 1

Minimal error

The error depends on the heterogeneity between the honest nodes



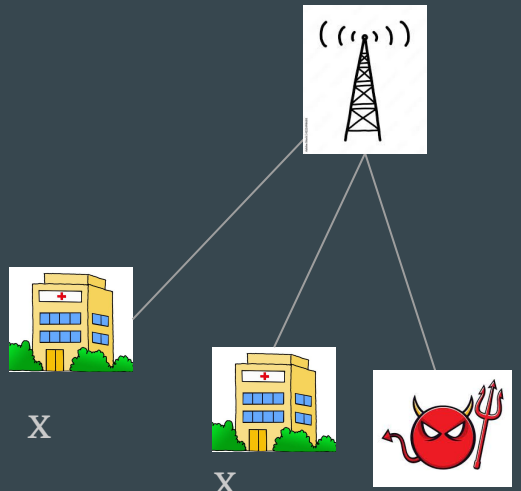
Setting 1



Setting 2

Minimal error

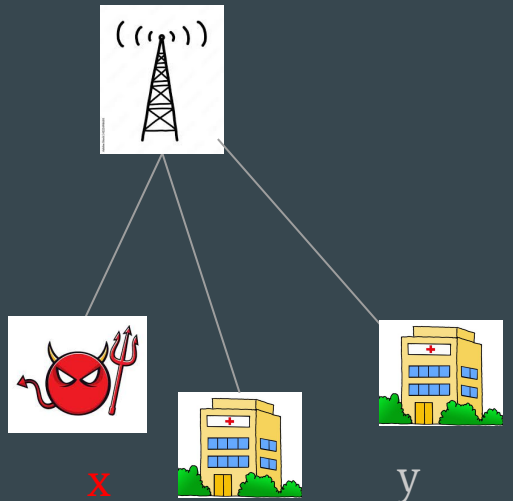
The error depends on the heterogeneity between the honest nodes



Setting 1

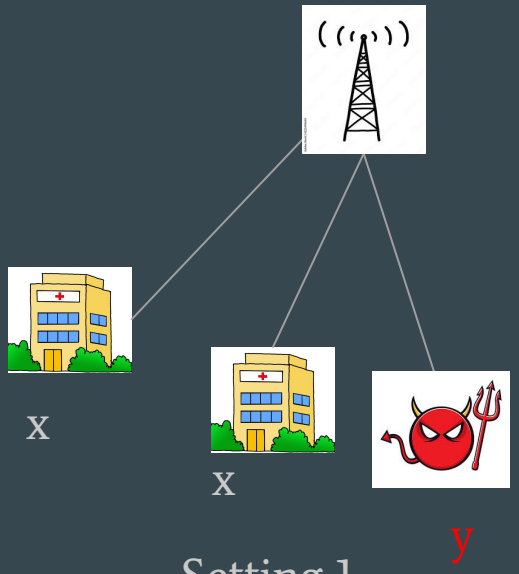
INDISTIGUISHABLE

Error depends on
| x - y |



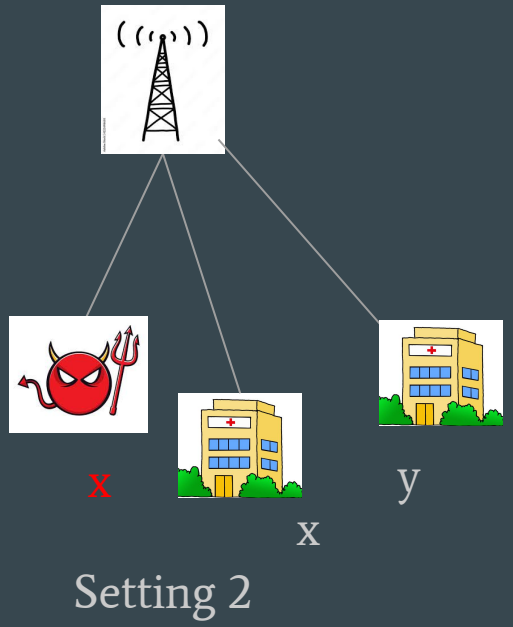
Setting 2

Minimal error



INDISTIGUISHABLE

Error depends on
| x - y |



r-robustness not possible in this case!

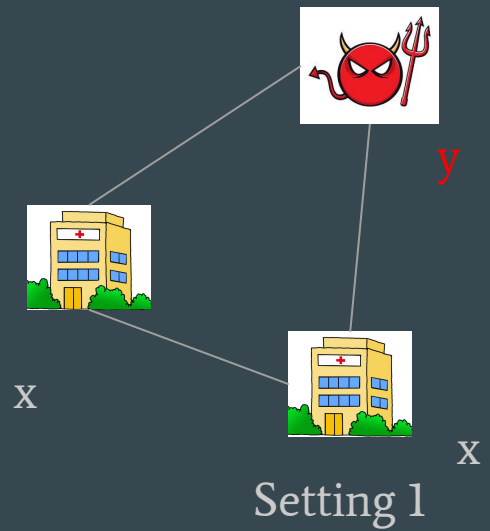
Decentralized setting

Maximal proportion of adversaries

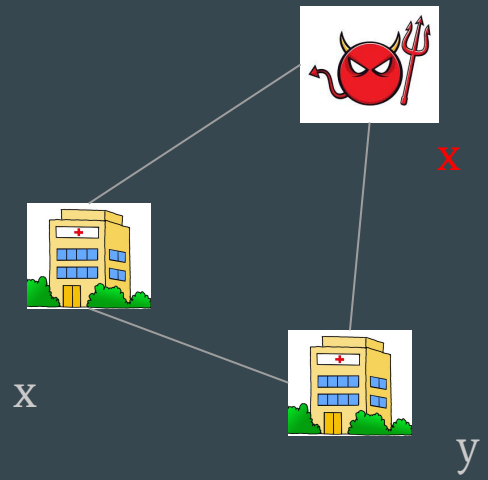
Decentralized: no robustness is possible if a third or more of byzantine nodes

Maximal proportion of adversaries

Decentralized: no robustness is possible if a third or more of byzantine nodes



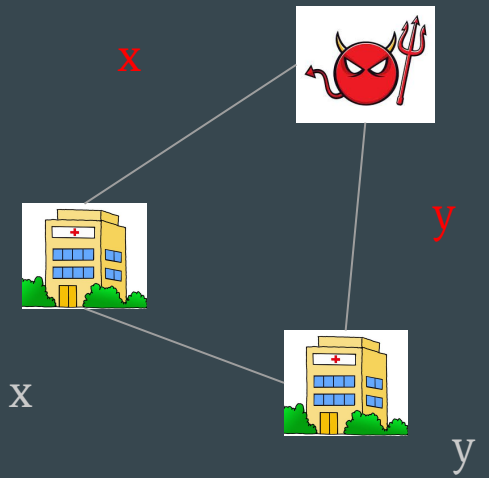
Setting 1



Setting 2

Maximal proportion of adversaries

Same Byzantine node can send different values to different nodes



It's the number of edges between honest and byzantine nodes that matters

Maximal proportion of adversaries

It’s the number of edges per node that matters, not the number of edges in total



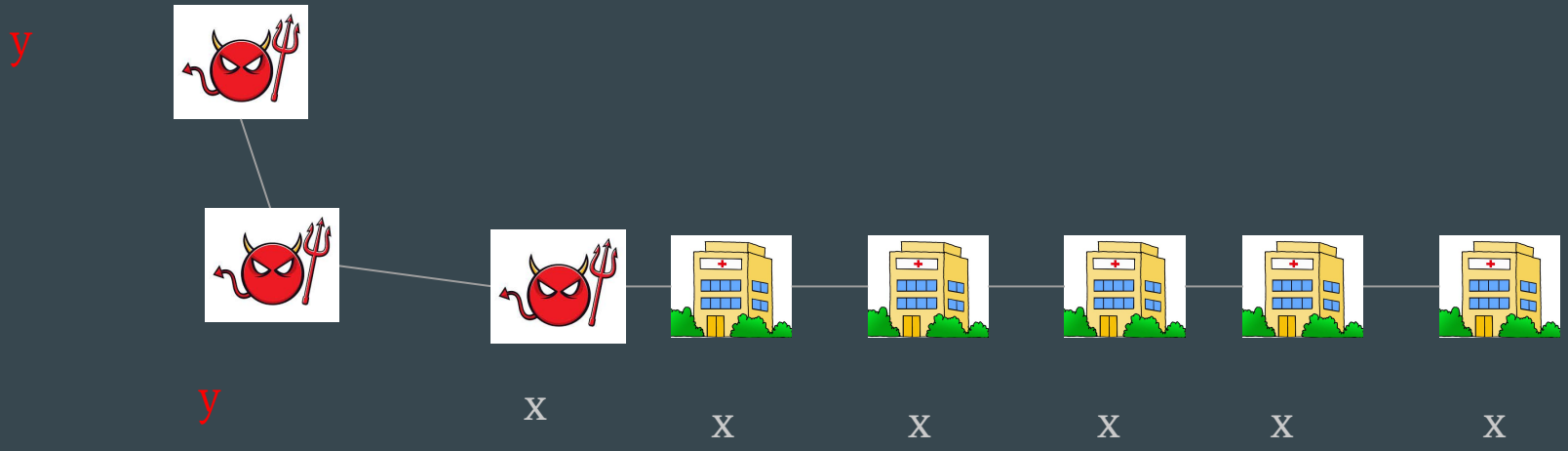
Maximal proportion of adversaries

It's the number of edges per node that matters, not the number of edges in total



Maximal proportion of adversaries

It’s the number of edges per node that matters, not the number of edges in total



Breakdown points

- Centralized communications:
    - ½ of Byzantine nodes (arbitrary changes)
    - ⅓ of Byzantine nodes (r-robustness)

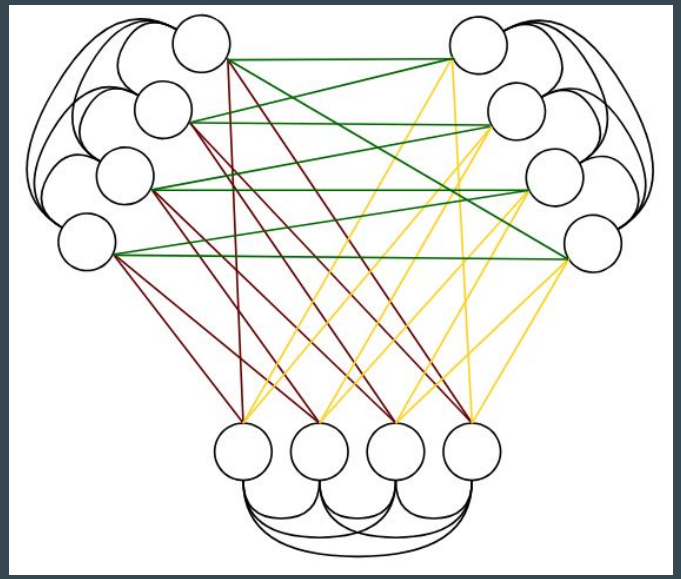
- Decentralized communications: ⅓ of Byzantine nodes

Breakdown points

- Centralized communications:
  - ½ of Byzantine nodes (arbitrary changes)
  - ⅓ of Byzantine nodes (r-robustness)

- Decentralized communications: ⅓ of Byzantine nodes
  - Breaks for 1 Byzantine node on a 3 nodes complete graph.
  - No Link between the breakdown point and the connectivity of the graph

Extending the 3-nodes construction



3 cliques

Each node is connected with b other nodes from the other cliques

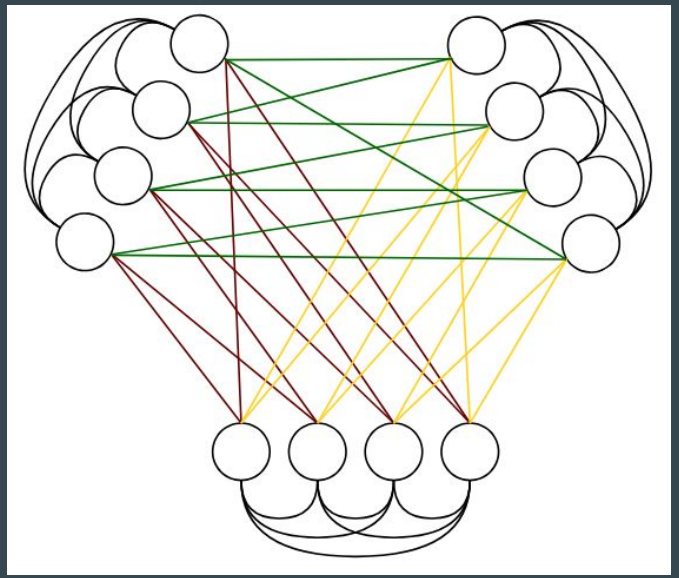
One clique is made only of Byzantine nodes

Algebraic connectivity of the honest subgraph

$$\mu_2 = 2b$$

(spectral gap of the Laplacian)

Breakdown point in arbitrary graphs



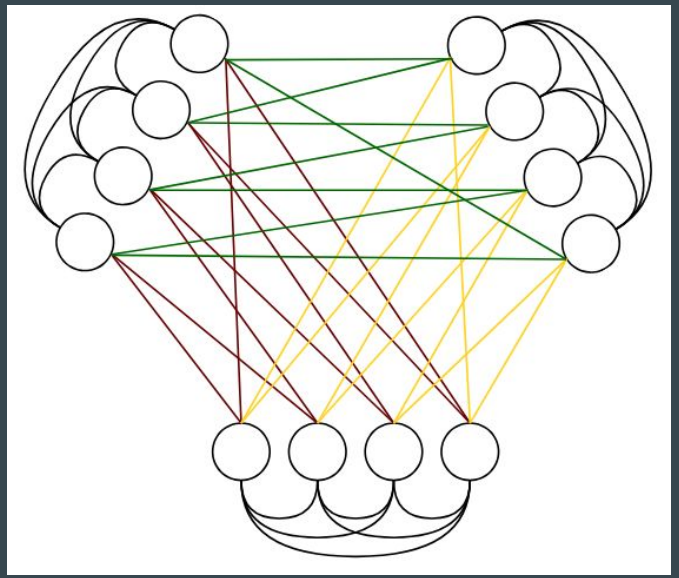
Let b be the max number of Byzantine neighbours honest nodes can have.

Thm (informal)
For any $h \geq 2b$, there exists a graph with h honest nodes and algebraic connectivity $\mu_2 = 2b$ on which no communication algorithm is robust

Robust algorithms on arbitrary graphs require

$$2b < \mu_2$$

Breakdown point in arbitrary graphs



Let b be the max number of Byzantine neighbours honest nodes can have.

Thm (informal)
For any $h \geq 2b$, there exists a graph with h honest nodes and algebraic connectivity $\mu_2 = 2b$ on which no communication algorithm is robust

Robust algorithms on arbitrary graphs require

$$2b < \mu_2$$

Complete graph: 2b < n - b => b < n/3: ok

Different gossip notations

Gossip matrix W $\qquad W1 = 1$

Unnormalized Laplacian of the communication graph

$$L_W = \sum_{\text{edges } ij} (e_i - e_j)(e_i - e_j)^\top \qquad L_W 1 = 0$$

Second smallest eigenvalue μ_2

$$W = I - \alpha L_W \qquad \lambda_2 = 1 - \alpha \mu_2$$

A common choice is $\alpha = 1/\mu_{\max}$

Can this lower bound be matched?

Local averaging: gossip communications

Basic version: nodes average with their neighbours. Fancy version:

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} \omega_{ij} (x_t^{(i)} - x_t^{(j)})$$

Local averaging: gossip communications

Basic version: nodes average with their neighbours. Fancy version:

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} \omega_{ij} (x_t^{(i)} - x_t^{(j)})$$

Applying weighted gradient differences that preserve the average

$$X_{t+1} = X_t - \eta L_W X_t$$

Typically, W is the (weighted) Laplacian of the graph.

Robustifying gossip through clipping

Robustifying gossip:

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} \omega_{ij} \mathrm{Clip}(x_t^{(i)} - x_t^{(j)})$$

$$\mathrm{Clip}(x, \tau) = x \min\left(1, \frac{\tau}{\|x\|}\right)$$

$\tau_t^{(i)}$ is chosen such that a total weight of 2b is clipped.

Robustifying gossip through clipping

Robustifying gossip:

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} \omega_{ij} \mathrm{Clip}(x_t^{(i)} - x_t^{(j)})$$

$$\mathrm{Clip}(x, \tau) = x \min\left(1, \frac{\tau}{\|x\|}\right)$$

$\tau_t^{(i)}$ is chosen such that a total weight of 2b is clipped.

Works for $4b \leq \mu_2$: **near-optimal breakdown**

Optimal under oracle clipping rule...

Convergence results

$$\delta = \frac{4b}{\mu_2}$$

$$\mathrm{Var}_{\mathcal{H}}(\boldsymbol{x}^t) \leq (1-\gamma(1-\delta))^t \,\mathrm{Var}_{\mathcal{H}}(\boldsymbol{x}^0),$$

$$\|\overline{\boldsymbol{x}}^t_{\mathcal{H}} - \overline{\boldsymbol{x}}^0_{\mathcal{H}}\| \leq \frac{\sqrt{\gamma\delta}\left(1-[1-\gamma(1-\delta)]^{t/2}\right)}{1-\sqrt{1-\gamma(1-\delta)}}\sqrt{\mathrm{Var}_{\mathcal{H}}(\boldsymbol{x}^0)}.$$

When $t \to \infty$, we have that $\mathrm{Var}_{\mathcal{H}}(\boldsymbol{x}^t) \to 0$ (so, consensus is reached) and:

$$\|\overline{\boldsymbol{x}}^t_{\mathcal{H}} - \overline{\boldsymbol{x}}^0_{\mathcal{H}}\|^2 \leq \frac{\gamma\delta}{(1-\sqrt{1-\gamma(1-\delta)})^2}\mathrm{Var}_{\mathcal{H}}(\boldsymbol{x}^0) \leq \frac{4\delta}{\gamma(1-\delta)^2}\mathrm{Var}_{\mathcal{H}}(\boldsymbol{x}^0).$$

- Similar to standard gossip unless close to breakdown ($\delta \approx 1$)
- Plug-in results exist for “general” decentralized optimization
- Trade-off bias for variance

Beyond clipping: robust aggregations

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} w_{ij} \mathrm{Clip}(x_t^{(i)} - x_t^{(j)}, \tau_t^{(i)})$$

Just need to verify a “robust mean” property. $F\big((\omega_i, z_i)_{i \in [n]}\big) \approx \sum_{i \in S} \omega_i z_i$

Beyond clipping: robust aggregations

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} \omega_{ij} \mathrm{Clip}(x_t^{(i)} - x_t^{(j)})$$

Just need to verify a "robust mean" property. $F\big((\omega_i, z_i)_{i\in[n]}\big) \approx \sum_{i\in S} \omega_i z_i$

Close links with (f, κ)-robustness (Allouah et. al. (2023))

Beyond clipping: robust aggregations

$$x_{t+1}^{(i)} = x_t^{(i)} - \eta \sum_{j \in n(i)} \omega_{ij} \mathrm{Clip}(x_t^{(i)} - x_t^{(j)})$$

Just need to verify a "robust mean" property. $F\big((\omega_i, z_i)_{i\in[n]}\big) \approx \sum_{i\in S} \omega_i z_i$

Close links with (f,κ)-robustness (Allouah et. al. (2023))

$$\left\| F\big((\omega_i, z_i)_{i\in[n]}\big) - \sum_{i\in S} \omega_i z_i \right\|^2 \leq \rho b \sum_{i\in S} \omega_i \|z_i\|^2.$$

Satisfied for clipping, but also for trimming, (coordinate-wise) median, etc... with different values for ρ

Take-away

One can prove a tight lower bound on the breakdown point, with a simple almost matching algorithm (up to a factor 2), through a generic framework.

Don’t do robust averaging of parameters, do robust averaging of parameter differences (= dual updates).

Related work, perspectives

- Many interesting connections with:
    - NNA (very similar but trimming instead of clipping)
    - ClippedGossip (different choice of clipping threshold)

  In particular, we provide tight theory for these.

- Near-optimal: where does the mismatch come from? Lower or upper? Can we close this gap?

Conclusion

Distributed learning is a vast field

- Communication can either be a constraint or an opportunity

- Many communication models (centralized vs decentralized, failures, etc...)

- Many nodes behaviour models (honest, honest but curious, adversarial...)

- Depends on the setting, but also critically on the hardware that is used

Conclusion

- The distinction between Federated Learning and decentralized is often overplayed: as long as nodes have their own parameter, techniques are very similar

- Decentralized analyses are more complex, but if done properly they are very insightful (forces to think the problem through, tricks don’t work)

Conclusion

Because of all these aspects

- Precise bounds need to be derived to correctly take into account the complexity of these settings, and build theory that is predictive of practice.

- The complexity of ad-hoc analyses quickly grows out of hands: need to construct principled approaches to factor and abstract away the complexity.

- Decentralized/distributed optim is just regular optim on very specific problems, that would not be interesting otherwise.

Why do we do scientific research?

Atelier SEnS - Sciences, Environments, Societies

Do the results of our research contribute to building a world that corresponds to our values?

- Researchers responsibility
- (Non-)neutrality of science



Do artifacts have politics? Langdon Winner
https://sens-gra.gitlabpages.inria.fr/atelier-impacts-recherche/



The tomato harvester

Langdon Winner, Do artifacts have politics?



- Researchers drastically shaped society in a way they did not intend to
- They received little profit / academic consideration

- Developed by 2 researchers from University of California, on public funds.
- Developed less tasty but sturdier tomato species to further increase yields
- Very expensive for farmers (more than 1000 tons to make it worth).
- 4 000 => 600 growers in 15 years, 32 000 jobs eliminated
- Overall productivity increased, benefiting to large farmers at the detriment of rural communities

A lawsuit was filed, charging that University officials are spending tax money on projects that benefit a handful of private interests to the detriment of farmworkers, small farmers, consumers, and rural California generally

How hard was it to predict the outcome? Am I building a tomato harvester? Do I want to build a tomato harvester?