Consensus-Based Distributed Online Prediction and Stochastic Optimization

Michael Rabbat

Joint work with Konstantinos Tsianos

McGill

$\mathcal{O}(\sqrt{m})$ Regret with Approximate Distributed Mini-Batches

Michael Rabbat

Joint work with Konstantinos Tsianos

McGill

Stochastic Online Prediction

Predict	$w(1)$	$w(2)$	$w(3)\ldots$
Observe	$x(1)$	$x(2)$	$x(3)\ldots$
Suffer Loss	$f(w(1),x(1))$	$f(w(2),x(2))$	$f(w(3),x(3))$

Assume $x(t)$ drawn i.i.d. from unknown distribution

Regret: $$R(m)=\sum_{t=1}^{m} f(w(t),x(t))-\sum_{t=1}^{m} f(w^*,x(t))$$

where $$w^*=\arg\min_{w\in\mathcal{W}} \mathbb{E}_x[f(w,x)]$$

See, e.g., S. Shalev-Shwartz, "Online Learning and Online Convex Optimization, FnT 2012.

Problem Formalization

Assume: $f(w, x)$ is convex in w for all x

$$|f(w,x) - f(w',x)| \leq L\|w - w'\| \qquad \text{(Lipschitz continuous)}$$

$$\|\nabla f(w,x) - \nabla f(w',x)\| \leq K\|w - w'\| \qquad \text{(Lipschitz cont. gradients)}$$

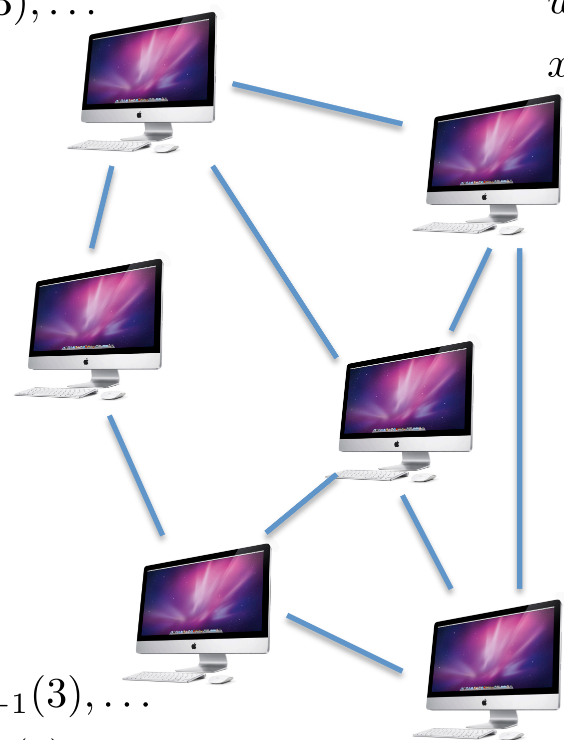
$$\mathbb{E}\left[\|\nabla f(w,x) - \mathbb{E}[\nabla f(w,x)]\|^2\right] \leq \sigma^2 \qquad \text{(Bounded variance)}$$

Regret: $$R(m) = \sum_{t=1}^{m} f(w(t), x(t)) - \sum_{t=1}^{m} f(w^*, x(t))$$

Best possible performance (Nemirovsky & Yudin '83) is $\mathcal{O}(\sqrt{m})$

Achieved by many algorithms including Nesterov's Dual Averaging

Distributed Online Prediction



Communication topology given by graph $G = (V, E)$

Regret: $$R_n(m) = \sum_{i=1}^{n} \sum_{t=1}^{m/n} \left[f(w_i(t), x_i(t)) - f(w^*, x(t)) \right]$$

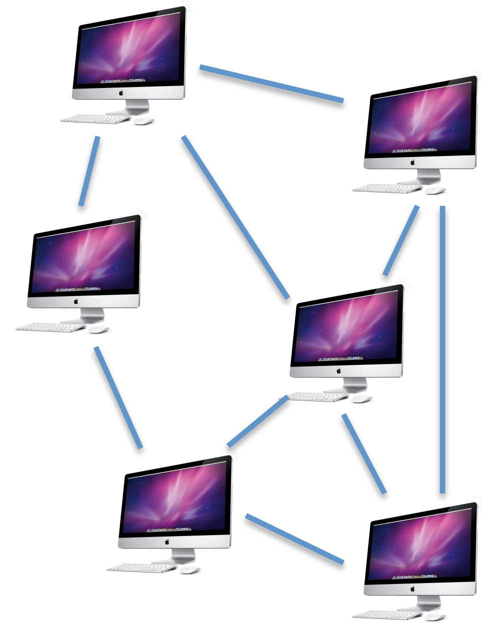
Distributed Online Prediction

"No collaboration"

Regret: $R_n(m) = nR_1(\frac{m}{n})$

$= \mathcal{O}(\sqrt{nm})$

With collaboration…



$R_n(m) \leq R_1(m) = \mathcal{O}(\sqrt{m})$

How to achieve this bound?

Mini-batch Updates

[O. Dekel, R. Gilad-Bachrach, O. Shamir, L. Xiao, JMLR 2012]

Predict	$w(1)$	$w(1)$	$\dots\ w(b) = w(1)$	$w(b+1)$
Observe	$x(1)$	$x(2)$	$\dots\ x(b)$	$x(b+1)$
Suffer Loss	$f(w(1), x(1))$	$f(w(1), x(2))$	$\dots\ f(w(1), x(b))$	$f(w(b+1), x(b+1))$

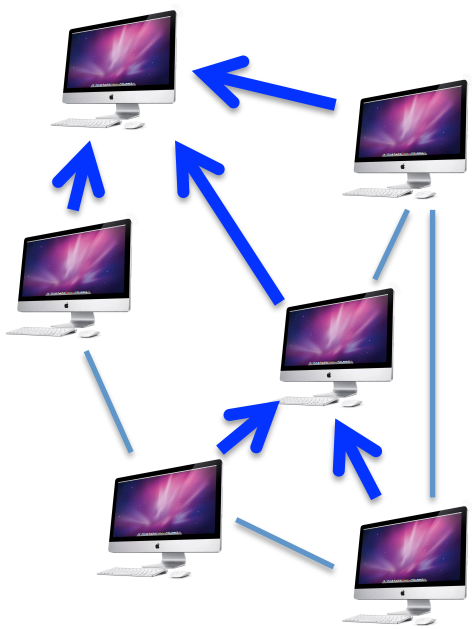
Update after mini-batch of b samples

Update using average gradient $\frac{1}{b}\sum_{t=1}^{b} \nabla f(w(1), x(t))$

Regret: $\mathbb{E}[R_1(m)] = \mathcal{O}(b + \sqrt{m+b})$

Distributed Mini-Batch Algorithm

[O. Dekel, R. Gilad-Bachrach, O. Shamir, L. Xiao, JMLR 2012]



Distribute each mini-batch of b samples across n nodes

Aggregate (synchronously) along a spanning tree (e.g., using ALLREDUCE)

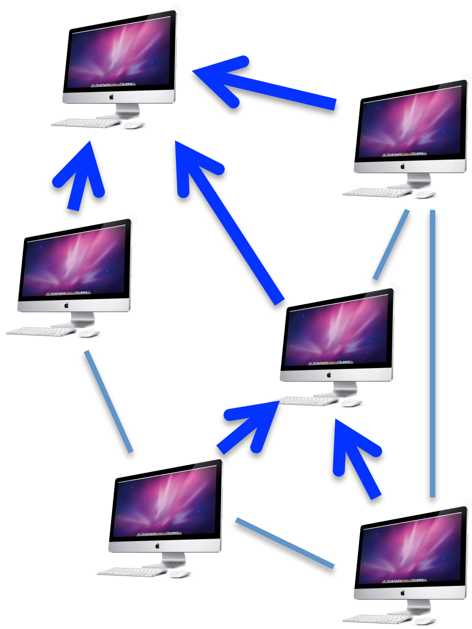
All nodes exactly compute $\frac{1}{b}\sum_{t=1}^{b} \nabla f(w(1), x(t))$

Collaborating has latency μ samples

Regret: $$R_n(m) = \sum_{t=1}^{\frac{m}{b+\mu}} \sum_{i=1}^{n} \sum_{s=1}^{\frac{b+\mu}{n}} [f(w_i(t), x_i(t,s)) - f(w^*, w_i(t,s))]$$

Distributed Mini-Batch Algorithm

[O. Dekel, R. Gilad-Bachrach, O. Shamir, L. Xiao, JMLR 2012]



Distribute each mini-batch of b samples across n nodes

Aggregate (synchronously) along a spanning tree (e.g., using ALLREDUCE)

All nodes exactly compute $\frac{1}{b}\sum_{t=1}^{b} \nabla f(w(1), x(t))$

Achieve optimal regret $\mathbb{E}[R_n(m)] = \mathcal{O}(\sqrt{m})$

with appropriate choice of b

Is exact average gradient computation necessary?

Can we achieve the same rates with asynchronous distributed algorithms?

Approximate Distributed Averaging

ALLREDUCE is an example of an *exact* distributed averaging protocol

$$y_i^+ = \textsc{AllReduce}(y_i/n, i) \equiv \frac{1}{n}\sum_{i=1}^{n} y_i \quad \forall i$$

Approximate Distributed Averaging

ALLREDUCE is an example of an *exact* distributed averaging protocol

$$y_i^+ = \textsc{AllReduce}(y_i/n, i) \equiv \frac{1}{n}\sum_{i=1}^{n} y_i \quad \forall i$$

More generally, consider approximate distributed averaging protocols

$$y_i^+ = \textsc{DistributedAverage}(y_i, i)$$

which guarantee that for all i

$$\|y_i^+ - \tfrac{1}{n}\sum_{i=1}^{n} y_i\| \leq \delta$$

with latency μ

Gossip Algorithms

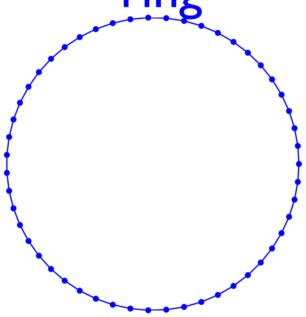
For a doubly-stochastic matrix W with $W_{i,j} > 0 \Leftrightarrow (i,j) \in E$ consider (synchronous) linear iterations

$$y_i(k+1) = W_{i,i} y_i(k) + \sum_{j=1}^{n} W_{i,j} y_j(k)$$

Then $y_i(k) \to \bar{y} \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} y_i(0)$ and $\|y_i(k) - \bar{y}\| \leq \delta$ if

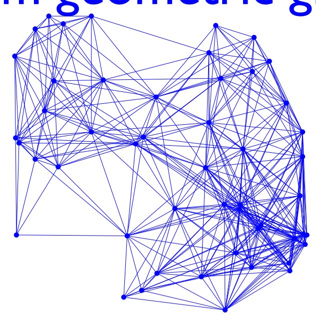
$$k \geq \frac{\log\left(\frac{1}{\delta} \cdot \sqrt{n} \cdot \max_j \|y_j(0) - \bar{y}\|\right)}{1 - \lambda_2(W)}$$

ring



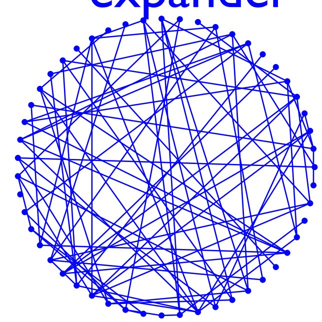
$$\frac{1}{1-\lambda_2} = \mathcal{O}(n)$$

random geometric graph



$$\frac{1}{1-\lambda_2} = \mathcal{O}\left(\sqrt{\frac{n}{\log(n)}}\right)$$

expander



$$\frac{1}{1-\lambda_2} = \mathcal{O}(1)$$

Gossip Algorithms

For a doubly-stochastic matrix W with $W_{i,j} > 0 \Leftrightarrow (i,j) \in E$ consider (synchronous) linear iterations

$$y_i(k+1) = W_{i,i} y_i(k) + \sum_{j=1}^{n} W_{i,j} y_j(k)$$

Then $y_i(k) \to \overline{y} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} y_i(0)$ and $\|y_i(k) - \overline{y}\| \le \delta$ if

$$k \ge \frac{\log\left(\frac{1}{\delta} \cdot \sqrt{n} \cdot \max_j \|y_j(0) - \overline{y}\|\right)}{1 - \lambda_2(W)}$$

Related work:

- Tsitsiklis, Bertsekas, & Athans 1986
- Nedic & Ozdaglar 2009
- Ram, Nedic, & Veeravalli 2010
- Duchi, Agarwal, & Wainwright 2012

Distributed Dual Averaging with Approximate Mini-Batches (DDA-AMB)

Initialize $z_i(1) = \mathbf{0}, w_i(1) = \mathbf{0}$

For $t = 1, \ldots, T \stackrel{\text{def}}{=} \lceil \frac{m}{b+\mu} \rceil$

$$g_i(t) = \frac{n}{b} \sum_{s=1}^{b/n} \nabla f(w_i(t), x_i(t,s))$$

$$z_i(t+1) = \textsc{DistributedAverage}(z_i(t) + g_i(t), i)$$

$$w_i(t+1) = \arg\min_{w \in \mathcal{W}} \left\{ \langle z_i(t+1), w \rangle + \beta(t) h(w) \right\}$$

Algorithm parameters
$0 < \beta(t) \leq \beta(t+1)$

Strongly convex prox function

Distributed Dual Averaging with Approximate Mini-Batches (DDA-AMB)

Initialize $z_i(1) = \mathbf{0}, w_i(1) = \mathbf{0}$

For $t = 1, \ldots, T \stackrel{\text{def}}{=} \lceil \frac{m}{b+\mu} \rceil$

$$g_i(t) = \frac{n}{b} \sum_{s=1}^{b/n} \nabla f(w_i(t), x_i(t,s))$$

b samples

$$z_i(t+1) = \textsc{DistributedAverage}(z_i(t) + g_i(t), i)$$

μ samples

$$w_i(t+1) = \arg\min_{w \in \mathcal{W}} \left\{ \langle z_i(t+1), w \rangle + \beta(t) h(w) \right\}$$

Should give $z_i(t+1) \approx \frac{1}{n} \sum_{i=1}^{n} \left(z_i(t) + g_i(t) \right)$

$$= \overline{z}(t) + \frac{1}{b} \sum_{i=1}^{n} \sum_{s=1}^{b/n} \nabla f(w_i(t), x_i(t,s))$$

When do Approximate Mini-Batches Work?

Theorem (Tsianos & MR): Run DDA-AMB with

$$k = \frac{\log\left((1 + 2L(b+\mu))\sqrt{n}\right)}{1 - \lambda_2(W)}$$

iterations of gossip per mini-batch and $\beta(t) = K + \sqrt{\frac{t}{b+\mu}}$, and take $b = m^\rho$ for $\rho \in (0, \frac{1}{2})$. Then $\mathbb{E}[R_n(m)] = \mathcal{O}(\sqrt{m})$.

If G is an expander, then $k = \Theta(\log n)$ and so $\mu = \Theta(\log n)$.
Latency is the same (order-wise) as aggregating along a tree.

Stochastic Optimization

Consider the problem

$$\begin{aligned} \text{minimize} \quad & F(w) = \mathbb{E}_x[f(w,x)] \\ \text{subject to} \quad & w \in \mathcal{W} \end{aligned}$$

Well-known that $F(\hat{w}(m)) - F(w^*) \leq \frac{1}{m}\mathbb{E}[R_1(m)]$

where $\hat{w}(m) = \frac{1}{m}\sum_{t=1}^{m} w(t)$

Distributed Stochastic Optimization

Corollary: Run DDA-AMB with $\beta(t) = K + \sqrt{\frac{t}{b}}$ and

$$k = \frac{\log\left((1 + 2Lb)\sqrt{n}\right)}{1 - \lambda_2(W)}$$

gossip iterations per mini-batch of b gradients processed across the network. Then

$$F(\hat{w}_i(\lceil \tfrac{m}{b} \rceil)) - F(w^*) = \mathcal{O}(\tfrac{1}{\sqrt{m}}) = \mathcal{O}(\tfrac{1}{\sqrt{nT}}) \ .$$

Accuracy $F(\hat{w}_i(T)) - F(w^*) \leq \epsilon$ is guaranteed if $T \geq \dfrac{1}{n} \cdot \dfrac{1}{\epsilon^2}$

Total gossip iterations: $\mathcal{O}\left(\dfrac{1}{\epsilon^2} \cdot \dfrac{\log n}{n} \cdot \dfrac{1}{1 - \lambda_2(W)}\right)$

Agarwal & Duchi (2011) obtain similar rates with an asynchronous master-worker architecture

Conclusions

- Exact averaging is not crucial for $\mathcal{O}(\sqrt{m})$ regret with distributed mini-batches
  - Just need to ensure nodes don't drift too far apart
- Current gossip bounds are worst-case in initial condition
  - Potentially use an adaptive rule to gossip less
- Fully asynchronous version is straightforward extension

- Open problem:
  - Does same approximate mini-batch approach extend to strongly-convex objectives?