Distributed Dual Averaging in Networks

Project Presentation

Maxim Timchenko

Electrical and Computer Engineering Department
Boston University

EC 719 Statistical Pattern Recognition, Fall 2014

The Paper

Agarwal, Alekh, Martin J. Wainwright, and John C. Duchi. "Distributed dual averaging in networks." In Advances in Neural Information Processing Systems, pp. 550-558. 2010.

Outline

Problem Statement

Decentralized optimization

Optimize a global objective formed by a sum of convex functions using local computation and local communication over a network of compute nodes.

Problem Statement

Decentralized optimization

Optimize a global objective formed by a sum of convex functions using local computation and local communication over a network of compute nodes.

Theoretical performance

Provide bounds on algorithm convergence rates as function of network size and topology.

Motivation: Big Data

Classical ML: minimize a loss function over a dataset.
Interesting datasets grow in size faster than innovations in storage capacity of a single computer.

- Google Maps dataset in 2012: 20 PB (20,500 TB)[1]
- Facebook dataset in 2010: 20 PB, in 2011: 30 PB[2]

Non-ML distributed convex minimization: multi-agent coordination, estimation in sensor networks, packet routing...

[1] http://mashable.com/2012/08/22/google-maps-facts/

[2] https://www.facebook.com/notes/paul-yang/moving-an-elephant-large-scale-hadoop-data-migration-at-facebook/10150246275318920

Motivation: Distributed Computation Constraints

In datacenter environments, supercomputers, and ad-hoc distributed networks, available bandwidth of communication is in inverse relationship to distance.

> *Traditionally, inter-cluster connectivity is oversubscribed, with much less bandwidth available between the clusters than within them. This assumes and then dictates that most intra-application communications occur inside the cluster."*[3]

Modelling the network as a graph with nearby connections as edges is convenient.

[3] https://code.facebook.com/posts/360346274145943/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/

Related Work

- N. Tsitsiklis, D. P. Bertsekas, and M. Athans. 1986. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *Uses shared memory.*
- D. P. Bertsekas, J. N. Tsitsiklis. Parallel and Distributed Computation: Numerical Methods. 1989.
- A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. 2009.
  Each agent has its own objective function.
- Y. Nesterov. Primal-dual subgradient methods for convex problems. 2009.
  Non-distributed version of the proposed algorithm.

Primal-dual subgradient methods

- Objective function $f(x)$ is non-smooth and not necessarily differentiable.
- Classic: $\min_x\{f(x) : x \in \mathbb{R}^n\} : x_{k+1} = x_k - \alpha_k g_k$
- Convergence and reach: $\alpha_k \to 0, \sum_k \alpha_k = \infty$.
- Lower linear model of the objective function:

$$l_k(x) \stackrel{def}{=} \sum_{i=0}^{k} \alpha_i (f(x_i) + \langle g_i, x - x_i \rangle) / \sum_{i=0}^{k} \alpha_i$$

- New subgradients are added with *decreasing* weights, needed for convergence of primal sequence $\{x_k\}$; but they are more important.

Primal-dual subgradient methods

- Insight: maintain two different sequences of parameters.
- One is responsible for the primal space process and has a vanishing sequence of steps.
- The other is in the dual space of linear function and has a non-decreasing sequence of weights to prioritize newer subgradients.

The Paper's Experimental Results

- The paper's simulations are done using synthetic SVM classification problems with hinge loss and

$$\mathcal{X} = \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 5\}.$$

- Performance is evaluated for different network sizes (n=100, 225, 400, 625, 900) and topology (single cycle, 2-D grid, bounded degree expander).
- "We have observed qualitatively similar behavior for other problem classes".

Our Experiments

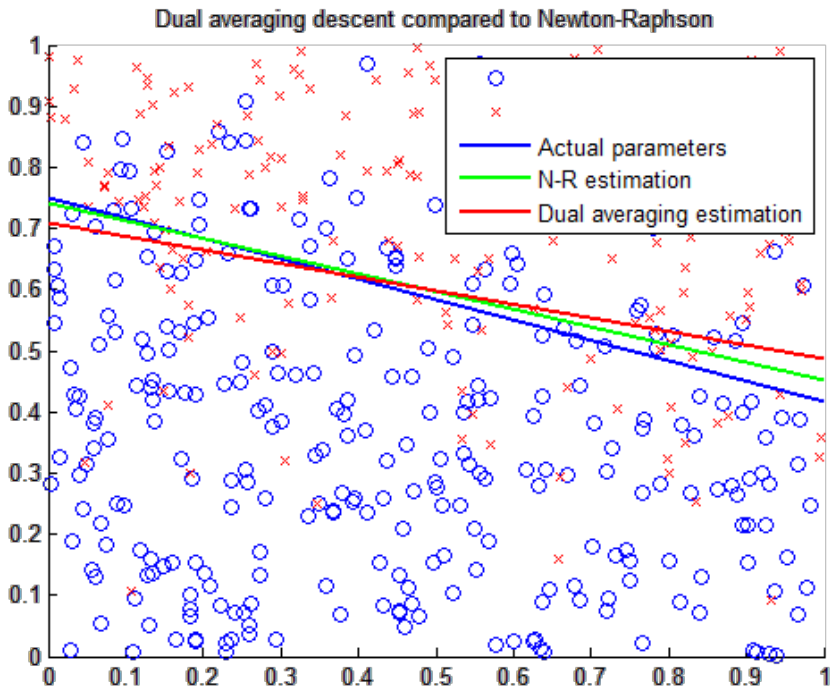
- Choose another class of optimization problems (non-SVM).
- Compare convergence rate of non-distributed primal-dual subgradient method to the proposed distributed method.
- Plot convergence rate vs. network size and parameters for a random geometric graph (mentioned in the paper but not evaluated).[4]
- Try the approach on a real dataset.

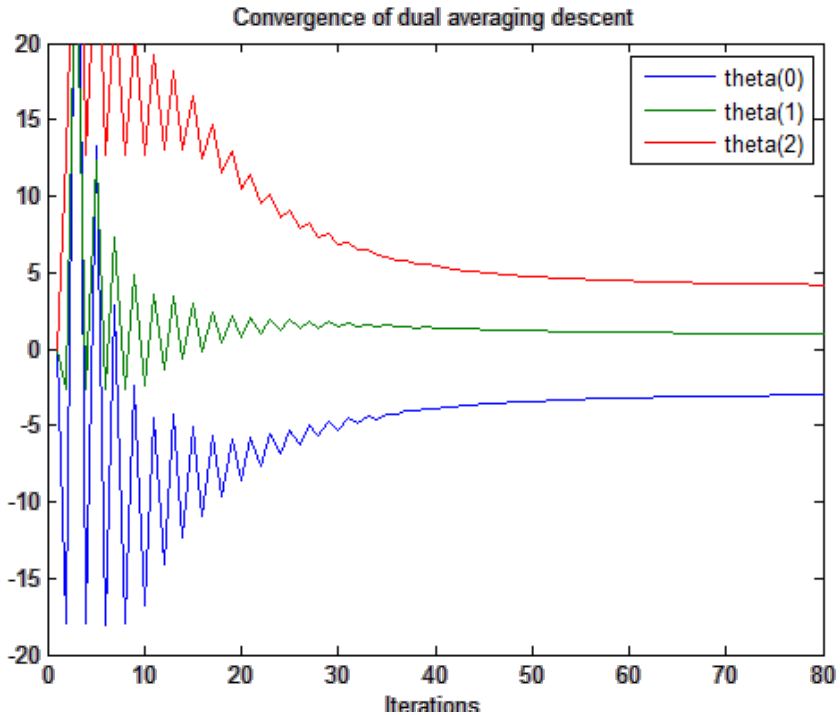
[4] A random geometric graph can model an ad-hoc wireless sensor network with limited communication range.

Experiments

- Implemented Dual Averaging for a 2D logistic regression: simple to understand, fast to simulate, easy to generate unlimited amount of test data.
- $f(\theta) = \min -\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})$
- $z(t+1) = z(t) - g(t) = z(t) - [\mathbf{x}(\mathbf{y} - h_\theta(\mathbf{x}))']$
- $\alpha(t) = 1/t, \psi(\theta) = \frac{1}{2}\|\theta\|^2$
- $\theta(t+1) = \Pi^{\psi}(-z(t+1), \alpha(t))$
- $\Pi^{\psi}(z, \alpha) = \text{argmin}_\theta(\langle z, \theta \rangle + \frac{1}{\alpha}\psi(\theta))$
- . . . a bit of algebra . . .
- $\theta(t+1) = -\frac{\alpha}{2} z$

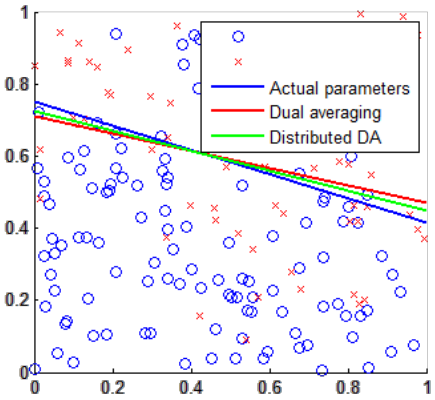
Single-node Dual Subgradient Method

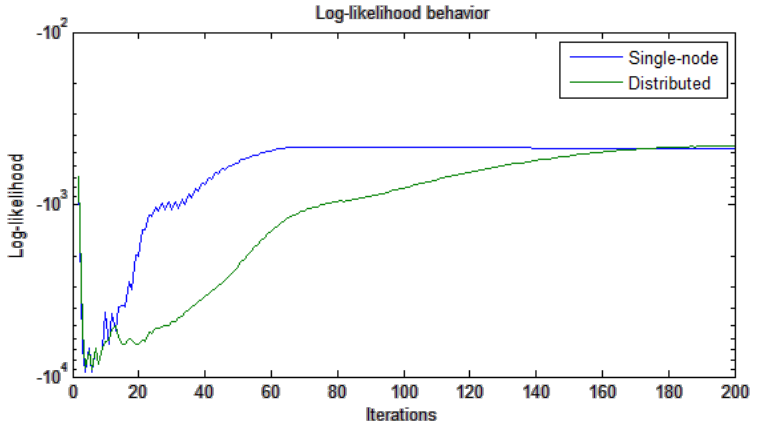


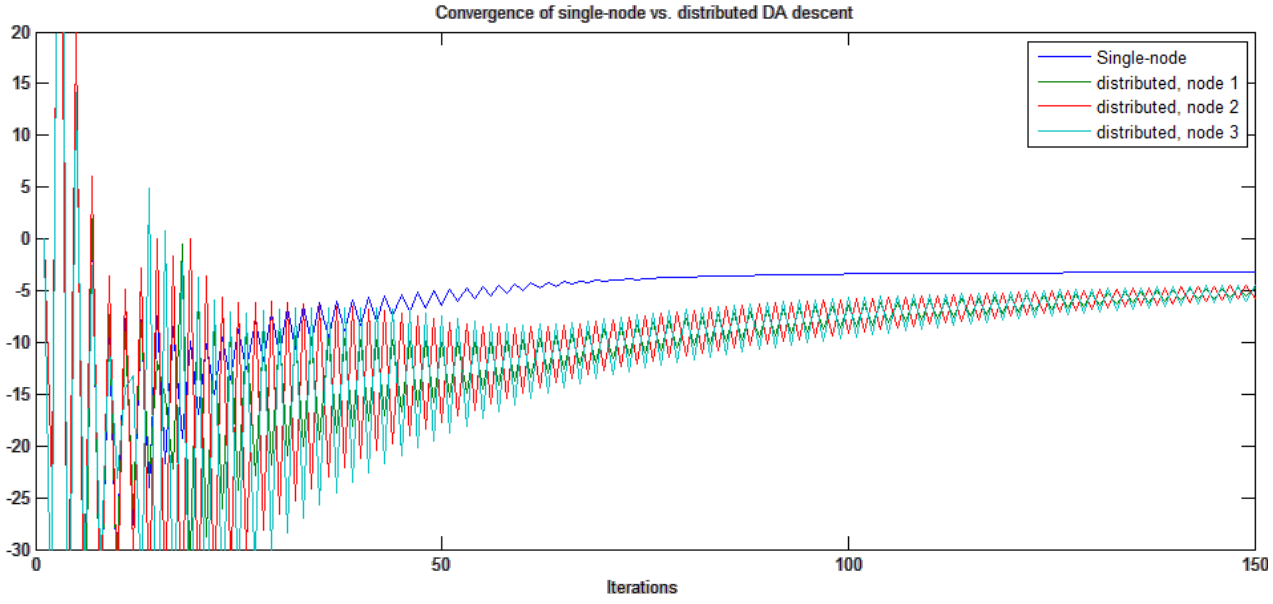


Compared to Newton-Raphson, convergence to same tolerance is much slower (258 iterations vs. 31)

Single-node vs. 9-node Cycle Distributed DA

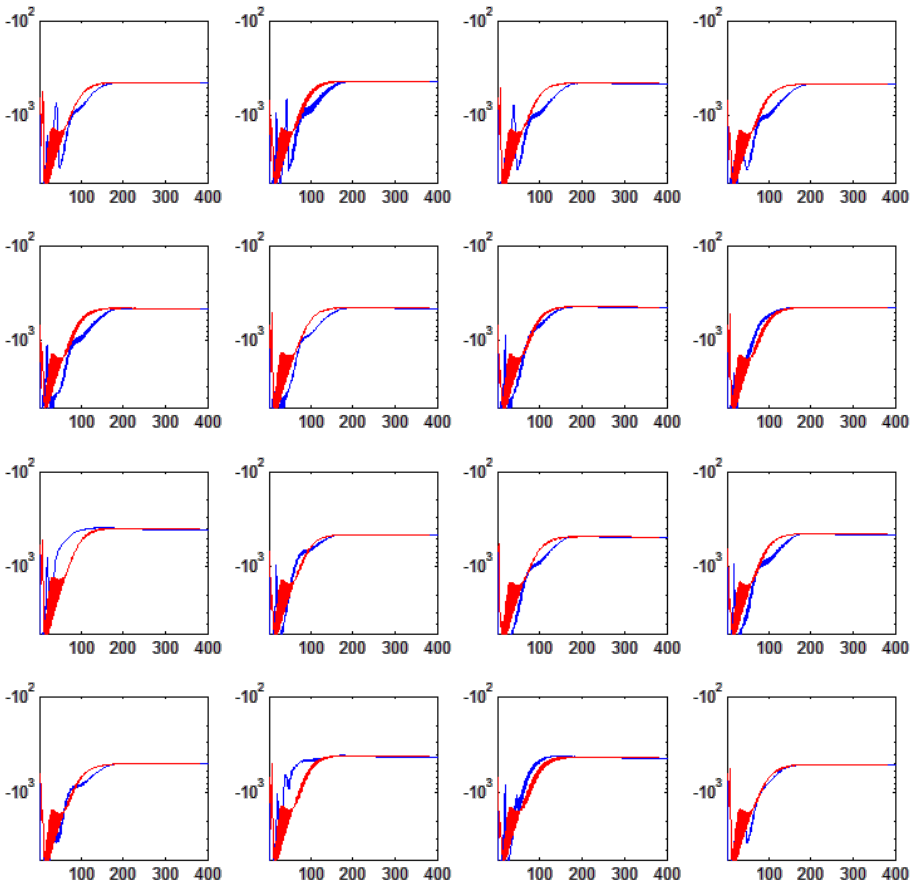


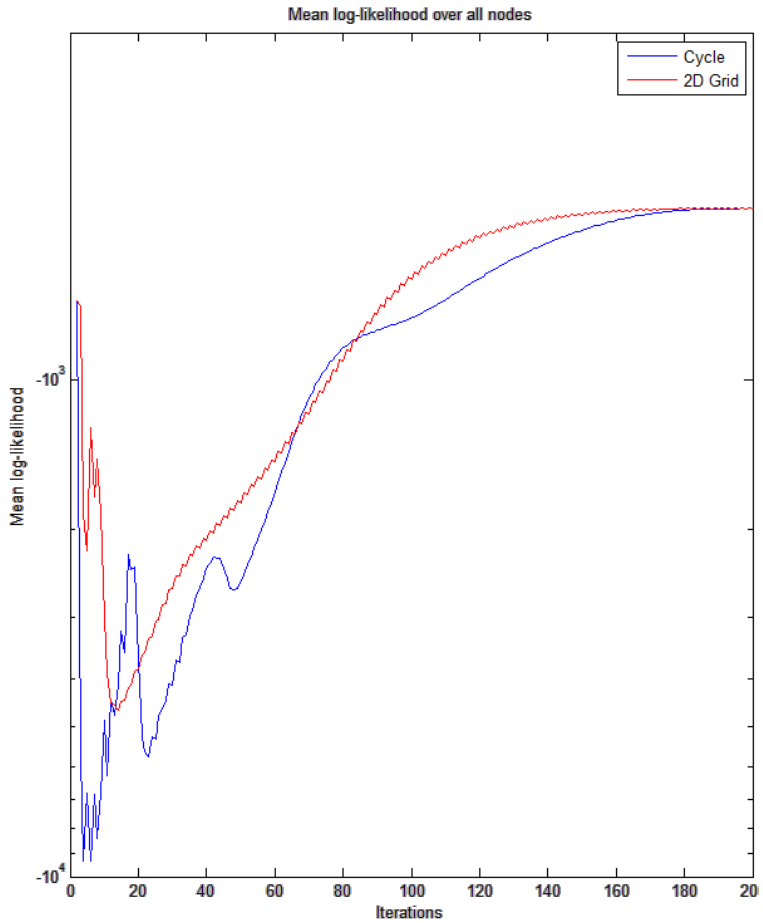




Iterations required: single-node = 261, distributed = 672.

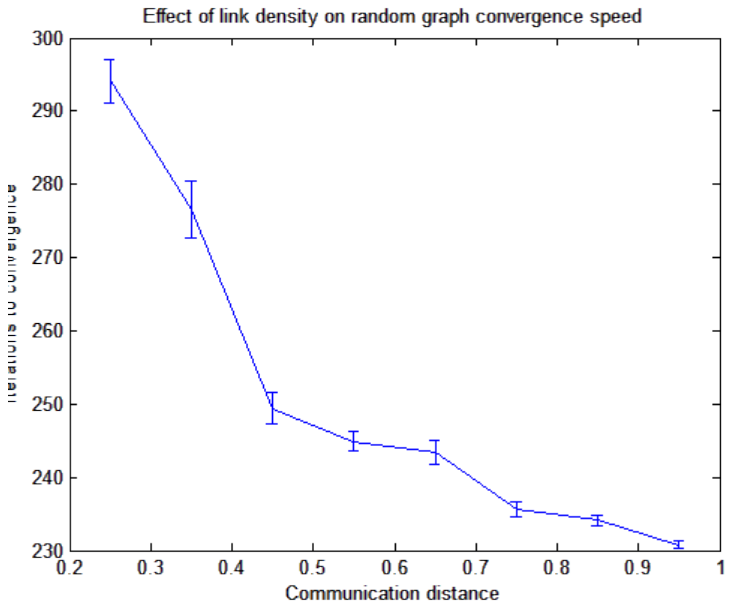
Convergence of a 16-node Cycle vs. 16-node Grid

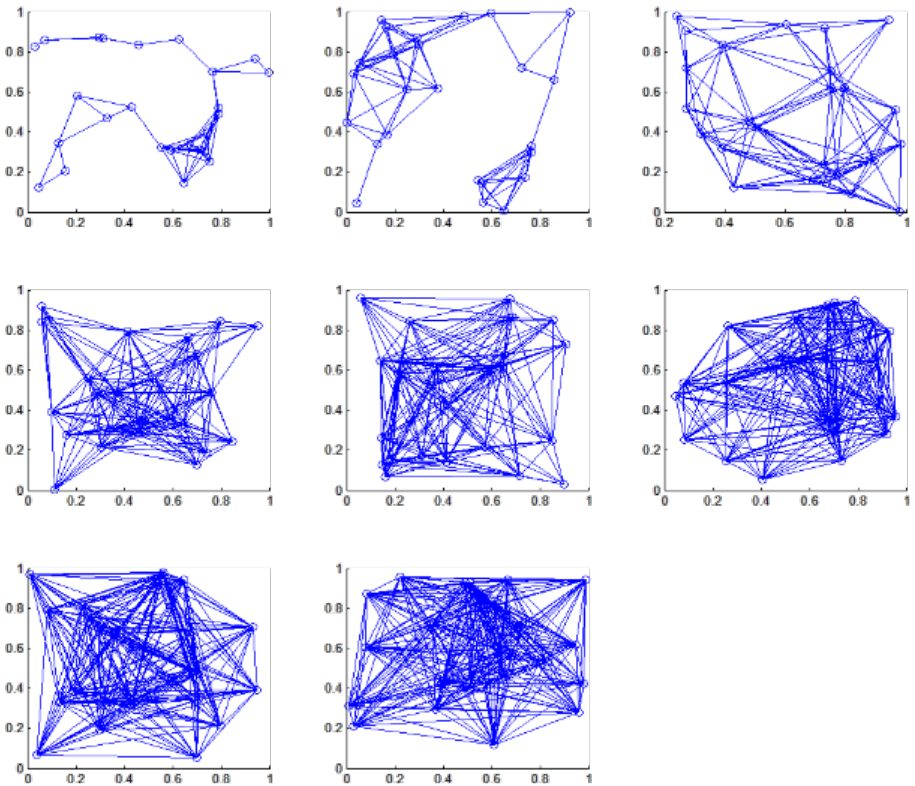




As connectivity improves, convergence speeds up: cycle graph iterations = 744, grid graph iterations = 383.

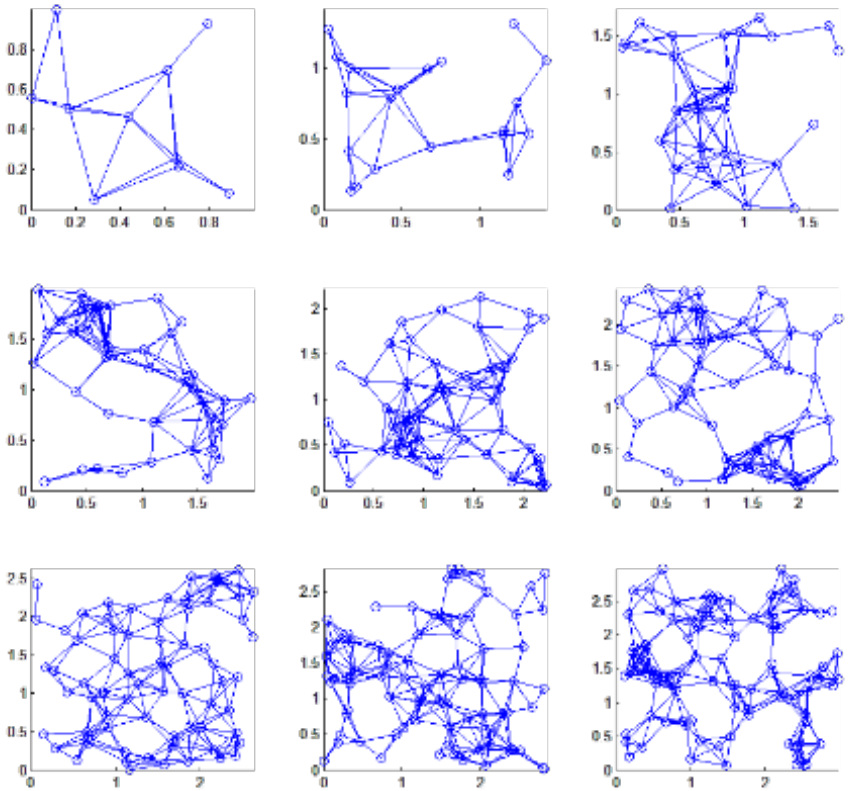
Random Geometric Graph: Density

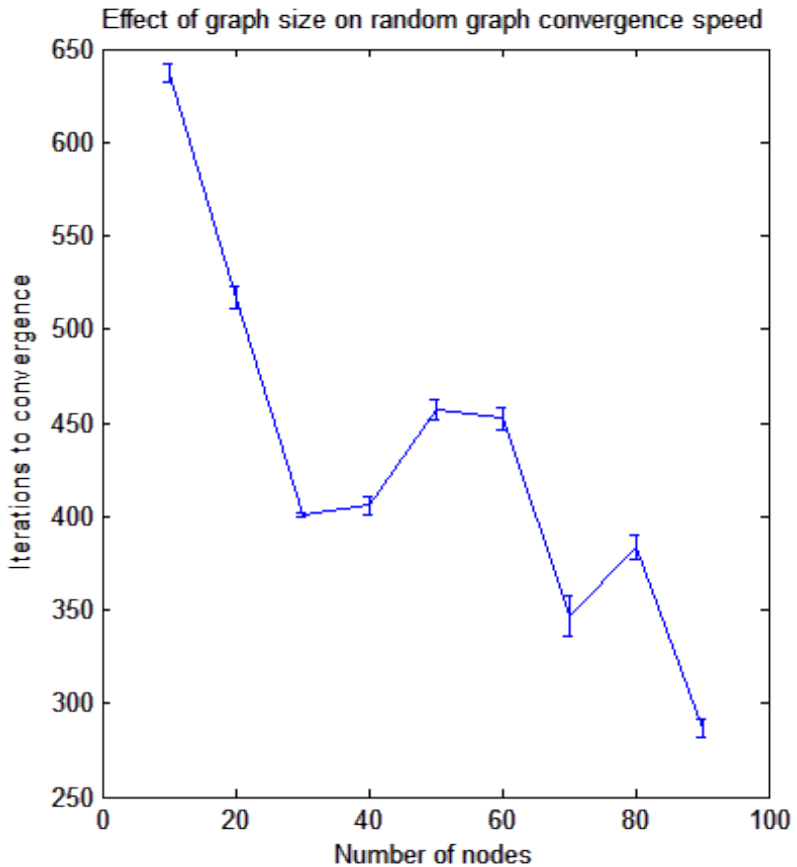




As expected, strong connectivity improves convergence speed.

Random Geometric Graph: Size (10-80 nodes)





The opposite result was expected; it looks like the effects of stronger connectivity outweigh the (quadratic) growth of area covered by the nodes of the graph.

Kernel SVM experiment

- Implement a kernelized SVM dual descent algorithm ("optimal soft margin linear classifier without offset" from HW2) on MNIST 4/9 8x8 dataset.
- Use kernel $@(a, b)(a' * b + 1)$ - fast but shows decent benchmark results with SMO.
- Inhomogenous polynomial kernel works well in a no-constant-term framework.
- The optimization problem is the SVM dual:

$$f(\alpha) = \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y^{(i)} y^{(j)} K(x^{(i)}, x^{(j)}) + \sum_i \alpha_i$$

- Optimal solution for z:

$$z_i = \frac{1 - \frac{1}{2} \sum_{i \neq j} \alpha_j y^{(i)} y^{(j)} K(x^{(i)}, x^{(j)})}{K(x^{(i)}, x^{(i)})}$$

Kernel SVM results

- After 500 iterations with $C = 1000$, $num\_sv = 1107$ and $test\_error = 0.064$.
- SMO result on the same kernel and C (normal / constant term framework): 0.045 with 234 SV.
- Conclusion: likely to be an implementation bug (the projection is now a constrained minimization problem, unlike logistic regression, since $0 \leq \alpha_i \leq \frac{C}{n}$).
- Dual averaging is only efficient when the projection from dual to primal space Π can be efficiently computed.

Conclusions: Tunable Parameters

- The details of the doubly stochastic graph matrix (for example, how much weight to assign to the local node?) influence convergence speed significantly but are not discussed.
- Step choice $\alpha(t)$ and convergence rate: with the suggested formula

$$\alpha(t) = \frac{R\sqrt{1-\sigma_2(P)}}{4L\sqrt{t}}$$

the convergence was extremely slow. Therefore we were unable to test the convergence corollaries for different graph types.

Conclusions: SVM

- Computing the projection is a more complicated task with a constrained problem and is likely an iterative optimization run in its own right.
- The goal of distributed computation in this case is not clear (what is the meaning of averaging of α_i's?)

Questions from the Audience

The Matlab code used to produce the output for this presentation and the presentation itself can be found on GitHub:

`https://github.com/maxvt/science/tree/master/ec719_machine_learning/distributed_dual_averaging`