Ioannis Mitliagkas — Research Statement

Vision: Making data analysis accessible to small organizations. This requires work on optimizing the individual components as well as automating complete machine learning pipelines.

Modern data-driven applications rely on powerful computational infrastructure and skilled data scientists, people with full command of known statistical guarantees and efficient computation techniques, to deliver analytical insights. Big organizations like web companies, retailers and government have the resources to attract experts and maintain powerful infrastructure. Social groups, NGOs and academic research units, on the other hand, lack funding and struggle to attract the people and maintain infrastructure. Brokers like DataKind connect data scientists to organizations of impact, but depend on experts volunteering their time while being aggressively recruited in industry. To make data analysis accessible, it is necessary that we bridge the gap between tools, human capital and infrastructure. Work towards this ambitious vision can be broken into three major thrusts:

1. **Optimize** popular tools to make them resource-efficient. This includes work on faster algorithms and systems that perform well with less computation and less data, lowering the cost of entry for small groups. A big part of my research effort goes into understanding the fundamental limits of popular techniques and coming up with resource-light alternative algorithms and systems.
2. **Automate** the tuning and assembly of data analytics pipelines to make tools usable by non-experts. Tuning commonly used methods, and complex systems like in deep learning, requires work on understanding their *computational and statistical behavior* and providing *data-dependent guarantees*—a major focus of my research. The automated assembly of large models and analytics pipelines further requires a deep understanding of the *dynamics and interactions in complex systems*; this is a direction I am interested in exploring further.
3. **Educate** the new generation of non-experts in the use of this friendlier generation of tools.

Research Overview

I am interested in the theory, algorithms and systems in the intersection of statistics and computation.

I aim to understand the theory and practical desiderata, and take a high-level look at entire ML pipelines, improving the various components to make a better overall system. My research touches upon themes of *optimization, analytical guarantees and tuning*, all necessary for bringing this vision about. It spans high-dimensional statistics, machine learning, optimization, statistical inference and large-scale distributed systems. I focus on extending our understanding of classic tools and optimize their use either through new theoretical guarantees or algorithmic and systems modifications that can deliver unexpected benefits.

New theoretical insights on classic tools. Methods like principal component analysis [8], PageRank [6], stochastic gradient descent (SGD) [2, 1] and Gibbs sampling [3] are ubiquitous, because they have proved their merit time and again. My work considers classic, massively deployed tools from a new perspective. In some cases it provides new, data-dependent guarantees [7, 3, 8], or reveals previously unknown interactions between algorithms and hardware [2]. Other times it introduces modifications to extend use cases, improve performance and reduce the resource footprint [6, 8]. Working on classic tools means research with a high potential for impact.

Implementation and Systems. Theoretical and algorithmic breakthroughs have a bigger impact when translated into a real system. I find that working on a system prototype is useful as a proof of concept, as a source of new technical challenges and research questions, and as an experimentation platform. In Omnivore [4], we used my recent theory [2] in our prototype deep learning system that is an order of magnitude faster than state-of-the-art systems. In [6], I modified GraphLab to improve its PageRank performance. In [7], I implemented and deployed our graph algorithm on hundreds of AWS nodes using MapReduce.

A relationship with industry and research labs is a valuable source of technical challenges and funding. My asynchrony work resulted into a joint project with Intel and NERSC on a planned submission to SC'17, aiming for the Gordon Bell prize. It also led to visits and discussions with nVidia, Google, MIT Lincoln Labs and Microsoft Research, giving rise to new theoretical questions. My deep learning work led to a collaboration with Daniel Rubin's Stanford Radiology lab, applying our technology to histopathology and bone-tumor identification tasks. My work on PageRank for large graphs was motivated by interactions with Teradata and booking.com.

Past and Current Work Highlights

My research balances work on theory and experimentation with prototype systems.

Summary. During my postdoctoral appointment, I discovered a new connection between asynchrony and momentum with direct implications for the performance and tuning of existing deep learning systems deployed by Google, Microsoft, Intel, nVidia and others. Our prototype system uses this theoretical understanding to train deepnets an order of magnitude faster than the state-of-the-art. I coauthored a paper dispelling a commonly held conjecture regarding the relative performance of different scan orders in Gibbs sampling and another one providing understanding on when frequent model averaging benefits parallel SGD.

During my Ph.D., I came up with a novel analysis and guarantees for Streaming PCA, designed new algorithms for finding dense subgraphs and deployed them on a cluster of hundreds of machines using MapReduce, extended the notion of typicality to give strong converse results for inverse problems, analyzed dependent random walks and extended the GraphLab codebase to support randomized algorithms, improving its PageRank performance by an order of magnitude.

Theory

My theoretical work spans optimization, learning and inference.

Asynchrony Induces Momentum [2]. In this paper I showed that running SGD in an asynchronous manner can be viewed as adding a momentum-like term to the SGD iteration. The result does not assume convexity of the objective function, so it is applicable to deep learning systems. An important implication is that tuning the momentum parameter is important when considering different levels of asynchrony, and that recent results by big groups like Google's TensorFlow missed this key optimization and report suboptimal results for asynchronous configurations. Finally, my theory suggests new ways of counteracting the adverse effects of asynchrony: for example, using *negative algorithmic momentum* can improve performance under high asynchrony. Our results have gained attention from Google, Microsoft, nVidia and Intel, as well as a number of national labs.

Scan order in Gibbs sampling is important [3]. Gibbs sampling is a Markov Chain Monte Carlo multivariate sampling technique that iteratively draws variables from their conditional distributions. There are two common scan orders: random and systematic scan. It has been conjectured that the mixing times (number of steps required to get an unbiased sample) of random scan and systematic scan do not differ by more than a logarithmic factor. We showed by counterexample that this is not the case, and proved that the mixing times do not differ by more than a polynomial factor under mild conditions. To prove these relative bounds, I introduced a method of augmenting the state space to study systematic scan using conductance.

Streaming PCA [8]. Known phase transition results suggest that in the *noisy setting* the number of samples required to recover principal components is $O(dimension)$. This means that batch algorithms require $O(dimension^2)$ memory and storage and motivates a *memory-limited, single-pass streaming* algorithm. My work was the first to provide an algorithm along with global convergence guarantees and tight characterization of the sample complexity for the streaming PCA problem. It is easily parallelizable[1] and can handle an overwhelming number of sample entry erasures [5]. It has been implemented by Julia and R developers in the StreamingPCA[2] and OnlinePCA[3] packages, which have been downloaded over a thousand times.

Information Theoretic Bounds and Learning Rankings [11, 10]. Fano's inequality, commonly used for lower bounds, yields what is called a *weak converse theorem*: if problem-specific necessary conditions are not met, then the probability of recovery error is bounded away from zero, for any method. Our work [11] introduced different methodology for providing *strong* converse theorems for general inverse problems: the probability of error is shown to converge to one—a guaranteed disaster. In [10] I provided tight achievability and strong converse results for learning a number of rankings, jointly from many users' pairwise preferences.

[1] `https://github.com/mitliagkas/pyliakmon`

[2] `https://libraries.io/github/eric-tramel/StreamingPCA.jl`

[3] `https://rdrr.io/cran/onlinePCA/man/bsoipca.html`

Implementation and Systems

My work includes prototype deep learning and large-scale graph processing systems.

Omnivore: tuning deep learning systems [4]. Using our prototype system, we studied the factors affecting training time in multi-device deep learning systems. We showed that in the single-node setting throughput can be improved by at least 5.5× over state-of-the-art systems on CPUs. We used our novel understanding of the interaction between system and optimization dynamics from my theory to provide an efficient hyperparameter optimizer and achieve performance 1.9× to 12× better than the fastest state-of-the-art systems.

Fast PageRank Approximations on Graph Engines [6]. For this VLDB paper I modified the GraphLab engine to allow for *randomized synchronization* between the graph nodes. This change reduces the communication requirements of the algorithm significantly. As a side-effect, random walks on the graph are not independent anymore, posing an analytical challenge. We demonstrated experimentally and analytically that our method gives a 7× to 10× speed-up over the state of the art (vanilla GraphLab).

Densest k-Subgraphs on MapReduce [7]. The problem of finding the densest subgraph of k nodes is NP-hard and hard to approximate. Using a low-rank approximation for the adjacency matrix, and leveraging combinatorial methods for quadratic optimization over low-rank spaces we get an efficient solver and *data-dependent* quality guarantees. Our theory asserted that in all of our experiments we achieved at least 70% of the optimum density. Using my MapReduce implementation we solved for graphs with billions of edges.

Future Work

I plan to pursue a mixture of theoretic and systems work to make data analysis efficient and simple to use. This involves work on understanding the theory and practical desiderata, taking a high-level look at entire ML pipelines, and improving the various components to make a better overall system.

Exact analysis for optimization. Much of the existing analysis and guarantees use a series of bounds that yield good-enough, or even tight results, but often obscure intuition. Nesterov's accelerated gradient method is a classic example. There are often overlooked insights hiding in popular methods, waiting to be discovered. Intuition can come from theory and reveal unexpected phenomena and prescribe strategies, like using negative momentum to compensate for the effects of asynchrony [2]. Using exact tools, like families of orthogonal polynomials, to analyze system and optimization dynamics is a promising direction with great expository value.

Targeted, optimized inference. Gibbs sampling successively simulates variables from the univariate conditionals of a multivariate target distribution. The principal degree of freedom there is the *scan*, the order in which each variable is sampled. In [3] we studied the relative efficacy of *systematic*, and *uniform random scans*. Non-uniform scans, however, can lead to more accurate inferences both in theory and in practice. This effect is particularly pronounced when certain variables are of greater inferential interest. Pipelines that include inference components can benefit by the development of efficient procedures for optimizing the scan order.

Automated tuning and assisted pipeline assembly. Beyond improvements in individual components, we need to study their interactions when used in common pipeline configurations. This knowledge can drastically reduce the hyperparameter space [4]. Simple ideas like back-propagating quality constraints (imposed by the user on the output) can create coupled constraints and provide the opportunity of pipeline-wide tuning. Taking things a step further, we can envision a system that helps non-experts with assisted model assembly. Starting with prescriptions for standard use cases, and using recent results and guarantees from adaptive data analysis, we can help the user iterate over a different pipelines until good results are achieved, while controlling overfitting.

I look forward to setting up my own research group and pursuing my vision and research agenda. I hope to impart to my research assistants the same impetus for discovery I got from my advisors. I am confident that my motivating vision will keep generating challenging and exciting research questions for my students, research that will ultimately be translated into useful contributions to society.

References

[1] J. Zhang, C. De Sa, I. Mitliagkas, and C. Ré. Parallel SGD: When does averaging help? *OptML, Workshop at NIPS 2016*, 2016.

[2] I. Mitliagkas, C. Zhang, S. Hadjis, and C. Ré. Asynchrony begets momentum, with an application to deep learning. *Proc. of Allerton Conf. on Communication, Control and Computing, Monticello, USA*, 2016.

[3] B. He, C. De Sa, I. Mitliagkas, and C. Ré. Scan order in gibbs sampling: Models in which it matters and bounds on how much. *NIPS*, 2016.

[4] S. Hadjis, C. Zhang, I. Mitliagkas, and C. Ré. Omnivore: An optimizer for multi-device deep learning on cpus and gpus. *Under Review*, 2016.

[5] I. Mitliagkas, C. Caramanis, and P. Jain. Streaming PCA with Many Missing Entries. *Preprint*, 2015.

[6] I. Mitliagkas, M. Borokhovich, A. Dimakis, and C. Caramanis. FrogWild! fast pagerank approximations on graph engines. *VLDB*, 2015.

[7] D. Papailiopoulos, I. Mitliagkas, A. Dimakis, and C. Caramanis. Finding dense subgraphs via low-rank bilinear optimization. In *ICML 2014*, pages 1890–1898, 2014.

[8] I. Mitliagkas, C. Caramanis, and P. Jain. Memory limited, streaming PCA. In *NIPS*, pages 2886–2894, 2013.

[9] I. Mitliagkas, N. D. Sidiropoulos, and A. Swami. Joint power and admission control for ad-hoc and cognitive underlay networks: Convex approximation and distributed implementation. *IEEE Transactions on Wireless Communications*, 2011.

[10] I. Mitliagkas, A. Gopalan, C. Caramanis, and S. Vishwanath. User rankings from comparisons: Learning permutations in high dimensions. In *Proc. of Allerton Conf. on Communication, Control and Computing, Monticello, USA*, 2011.

[11] I. Mitliagkas and S. Vishwanath. Strong information-theoretic limits for source/model recovery. In *Proc. of Allerton Conf. on Communication, Control and Computing, Monticello, USA*, 2010.

[12] I. Mitliagkas, N. D. Sidiropoulos, and A. Swami. Distributed joint power and admission control for ad-hoc and cognitive underlay networks. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 3014–3017. IEEE, 2010.

[13] I. Mitliagkas, N. D. Sidiropoulos, and A. Swami. Convex approximation-based joint power and admission control for cognitive underlay networks. In *Wireless Communications and Mobile Computing Conference, 2008. IWCMC'08. International*, pages 28–32. IEEE, 2008.