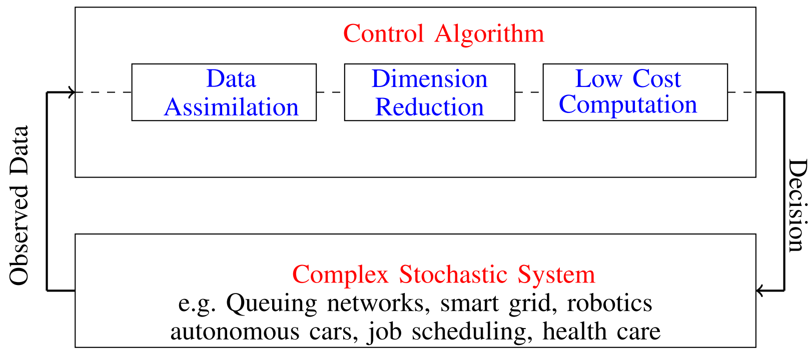
Research Statement
Algorithms for Stochastic Control

Chandrashekar Lakshminarayanan

I. Introduction

Stochastic Control (setup and goal): Real world systems are (i) uncertain: that is, there is randomness (e.g., the time taken to service a customer in a queue, or the return-on-investment in a particular stock), (ii) complex: too many configurations (such as possible positions in a game of chess, or possible configurations of vehicular traffic in a big city), (iii) dynamic: the system configuration changes with time (number of customers in a queues, the price of a stock). However, the dynamics of such systems also depends on the decisions that we make from time to time (e.g., investment portfolio, moving a chess-piece, traffic lights, etc). An immediate question is that of *optimal control* of such systems: can we come up with the right sequence of decisions?


Control Algorithm
Data Assimilation
Dimension Reduction
Low Cost Computation
Observed Data
Decision
Complex Stochastic System
e.g. Queuing networks, smart grid, robotics
autonomous cars, job scheduling, health care

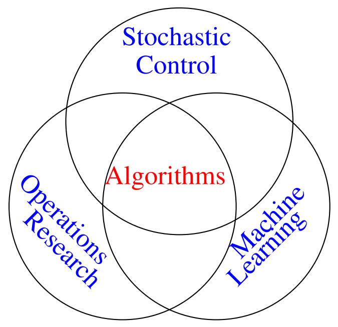


Fig. 1: On the left is a schematic of a data-driven control algorithm; the observed data is mapped to a decision rule. The challenge is to design control algorithms that are data efficient, computationally cheap and scalable. The right diagram shows the overlap between three areas. Stochastic control has been used to address problems in operations research in the past. A recent trend has been to used machine learning techniques to build data-driven control algorithms.

Control Algorithms (need and specification): As many applications are added by the day, and with the growing complexity, it is imperative to deploy algorithms to compute the decision rules (henceforth to be called as *policies*), an otherwise impossible task for solely analytical methods or heuristics. The schematic of a control algorithm is given in Section I, where the algorithm takes observations from the complex stochastic system as input and maps it to a policy. A natural question will then: what is desirable from such a control algorithm? what are the specifications? The following presents a non-exhaustive list.

- Scalability: The number of configurations of the system grows exponentially in the number of dimensions. It is desirable that the algorithms are computationally cheap, and are tractable.
- Data-Driven: In most scenarios, the underlying model parameters are not known and it becomes important for the algorithms to be solely data-driven. A closely related aspect is that of data efficiency, wherein the algorithms are expected to use all the available information in the given data samples.
- Performance Guarantee: The algorithms need to be stable, i.e., should produce convergent policies. Further, it is desirable that the computed policy is near-optimal.

Algorithm Design (tools and techniques): The schematic (Section I) shows the overall template of a control algorithm that is likely to meet the specifications of incurring low computational cost and of being data driven. While specific details of the algorithms could vary, all of them can be said to combine three keys components (not in any specific order) namely data assimilation, dimension reduction and low-cost computation. Markov decision processes (MDPs) is a useful tool to formulate stochastic control problems, and the Markovian nature gives rise to the idea of dynamic programming (DP) which forms the back bone of the computational procedure in most of such control algorithms. Ideas of related to dimensionality reduction and noise filtering come from theory of approximate dynamic programming and stochastic approximation respectively. Recent times have also seen the growth of a class of stochastic control algorithms called reinforcement learning (RL) algorithm which combine ideas from a variety of fields such as Markov decision processes (MDPs), approximate dynamic programming (to deal with dimensionality reduction) stochastic approximation (SA) (to deal with noisy data), and machine learning (to learn from sample trajectories). RL algorithms have been extremely successful in a variety of applications such as queuing networks, autonomous game play and robotics.

II. Past Work

In the previous section, we saw some specifications and design techniques for control algorithms to address the question of computing near-optimal policies. Naturally, the following question of interest: are computationally cheap data-driven methods stable? do they give useful policies? how fast do they estimate? I was fortunate to contribute towards the answering these questions during the past years.

ADP in $(\min,+)$ basis: A widely used dimensionality reduction technique is linear function approximation, wherein, the value function is approximated by the linear combination of set of chosen basis functions. By choosing fewer basis functions (in comparison to the size of the state space), the ADP algorithm is dimension free. One of the important issues with such ADP is that even the most fundamental methods such as approximate value iteration/policy iteration (AVI/API) don't have convergence guarantees, i.e., the computed sequence of policies can diverge. In our work [5], we showed that if the basis is chosen to be $(\min,+)$ linear (also known as tropical-linearity), then the ADP methods will be convergent and the performance of the computed policy depends on the function approximation.

Constraint reduction in approximate linear program: The approximate linear program (ALP) is an ADP method that has performance guarantees [7] for the computed policy and is guaranteed to converge (unlike other AVI/API). A limitation of the ALP is the intractably large number of constraints. In our work [8, 3], we provided theoretical guarantees for constraint reduction in ALP. Our arguments are based on geometry of the linear programs as opposed to previous works based on counting (VC-dimension) based arguments. We showed that the performance of ALP can be guaranteed by choosing the constraints corresponding to a *conic-cover* of the given basis functions.

Stability: We derived conditions that imply stability and boundedness of multi-timescale stochastic approximation (SA) algorithms. Our work significantly extended the prior work on stability of single timescale SA algorithms [2], and provided a blanket result [9], that covers a widely used class of reinforcement learning algorithms called actor-critic methods.

Finite-Time Performance: Temporal difference (TD) class of learning algorithms have been widely used to learn value functions of a given policy. Such value function learning is a sub-loop in many important RL algorithms (such as actor-critic methods). Many of the important TD algorithms are also linear stochastic (LSA) approximation methods. The choice of the stepsize or learning rate is critical for the performance of LSA algorithms. In our work [6], we studied LSA with constant stepsize and iterative averaging, and showed that TD algorithms can be deployed with a universal constant stepsize, and achieve a problem instance dependent $O(1/t)$ rate for the convergence of mean-squared estimation error. Our work eliminates stepsize tuning in a host of TD algorithms.

I was also fortunate to work of some interesting applications namely crowd-sourcing and Hadoop. In our work [4], we showed that learnt optimal pricing policies achieve better completion times for the tasks posted onto crowd. In our work [1], we proposed a noisy gradient algorithm to tune the parameters in Hadoop framework; our approach achieved 25% to 60% improvement in performance over the default parameter setting.

III. Future Work

I strongly believe that new practical application problems give rise to new design specifications and algorithmic ideas. Thus, I strive to strike a healthy balance between theoretical questions as well as practical problems in stochastic control.

Constrained and Risk-Sensitive Control: Several applications need policies to satisfy one or more constraints. For example, in energy harvesting sensor networks, each sensor is constrained by the amount of energy it can expend at any given time, or an autonomous driving system has speed limitations while navigation, or a stock investment policy cannot fluctuate the portfolios in a non-smooth manner. Another aspect is that of risk-sensitivity, i.e., the policies are not supposed to visit certain *dangerous* states. For example, while controlling a autonomous helicopter, there could be states that lead to instability and eventually cause the overall system to crash. Thus, developing reinforcement learning algorithms for risk-sensitive and constrained control problems is an interesting research direction with a variety of applications.

Exploration vs Exploitation: An important stochastic control problem is that of efficient data assimilation, i.e., based on prior data we need a policy that learns about the environment in an optimal way. An example application is health care, wherein, it is important to gather maximum patient information while making minimal possible tests. Also, in many applications the feedback might be complex, i.e., we can obtain the rewards only for a group of actions (by several decentralized control units), or the feedback is delayed. Typically, these problems are posed as multi-armed bandit problems, while the independent arms case and the linear MAB problems have been well understood in literature, newer applications typically require significant modification of the basic MAB algorithms. In particular, multi-agent systems such as sensor networks, or robots, decentralized MAB algorithms are required for effective co-ordination via optimal communication.

Classical vs Learning based Control: Classical control as well as learning based algorithms have the same objective, i.e., to produce a decision rules. However, the approaches in both the areas are quite different. Classical control is mostly model-based and learning based algorithms do not worry about the model, but instead are geared towards design of solely data-driven controllers. Specifically, several learning control algorithms have deep neural networks (DNNs) as their integral part, and several questions are still open: why do DNNs generalize? when are DNNs trainable? when do DNNs produce reliable policies. An interesting research direction is to use tools from control theory to answer some of the questions. For instance, several training algorithms for DNNs such as Nestrov's accelerated gradient have be interpreted as dynamical systems, and tools from robust control theory have been used for principled design of these algorithms. As we move forward to newer applications, developing and analysis hybrid controllers based on classical as well as learning techniques is of interest.

Vision and Control: A common thread connecting some of the recent success of learning based control such as Atari, AlphaGo, autonomous driving, is that the input is a vision signal. A general view is that vision signals are rich in representing the underlying environment and by learning the inherent symmetries in the vision input can help in efficient control. This leads to an interesting question in the other direction: can control policies also dictate attention towards particular parts in the field of view to gain useful information? To answer these questions, it is important to pose a combined vision and control problem to design newer algorithms.

IV. Conclusion

This document outlines some of my past and future research directions n the area of algorithms for stochastic control. These plans and directions are in various levels of concreteness; some of them require significant extension of existing theory, other require newer mathematical formulations. Most of these problems have well founded application areas as well. I also believe that these questions are inherently flexible to cater the interests of most students. The list of directions are not exhaustive and I will be constantly strive to add newer directions with the progress of time.

References

[1] V. S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint.* TRIM, 2008.

[2] S. Bhatnagar C. Lakshminarayanan and C. Szepesvári. A generalized reduced linear program for markov decision processes. *IEEE Transactions on Automatic Control*, 2017.

[3] L. Chandrashekar, A.Dubey, S. Bhatnagar, and B. Chithralekha. A markov decision process framework for predictable job completion times on crowdsourcing platforms. In *Proceedings of the Seconf AAAI Conference on Human Computation and Crowdsourcing, HCOMP 2014, November 2-4, 2014, Pittsburgh, Pennsylvania, USA*, 2014.

[4] L. Chandrashekar and S. Bhatnagar. Approximate dynamic programming with (min; +) linear function approximation for markov decision processes. In *53rd IEEE Conference on Decision and Control, CDC 2014, Los Angeles, CA, USA, December 15-17, 2014*, pages 1588–1593, 2014.

[5] L. Chandrashekar and C. Szepesvári. Linear stochastic approximation: How far does constant stepsize and iterate averaging go? *AISTATS*, 2018.

[6] D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[7] S. Kumar, S., L. Chandrashekar, P. Parihar, K. Gopinath, and S. Bhatnagar. Scalable performance tuning of hadoop mapreduce: A noisy gradient approach. In *2017 IEEE 10th International Conference on Cloud Computing (CLOUD), Honolulu, HI, USA, June 25-30, 2017*, pages 375–382, 2017.

[8] C. Lakshminarayanan and S. Bhatnagar. A generalized reduced linear program for markov decision processes. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2722–2728, 2015.

[9] C. Lakshminarayanan and S. Bhatnagar. A stability criterion for two timescale stochastic approximation schemes. *Automatica*, 2017.