

The scientific questions of our day are becoming increasingly complex. As datasets grow larger, analysts are no longer able to sift through them by hand. As control problems become more complicated, the human mind can no longer manage them. For example, how can a geologist mine volumes of seismic data and know where to drill for oil? How can a biostatistician analyze corpora of folded proteins to predict properties of new drugs? How can a roboticist construct good motor primitives for optimal control? Questions at the frontiers of every field require new levels of computing intelligence, and scientists and engineers will increasingly turn to the tools of machine learning to solve them.

To answer these questions, we can build **models of the world**—models of rock structures, models of proteins and molecular dynamics, or models of articulated robots. But as we grapple with more complex questions, these models are also becoming more difficult for humans to develop and use. The tools of machine learning are helping, but they need to be faster, more expressive and more flexible—and we must find ways to export them into the hands of non-specialists.

I study the mathematical and computational principles needed to build and use such models of the world, extending **machine learning**, **probabilistic modeling**, and **planning and decision making** to help address four key questions. First: how can we help the broader community use advanced machine learning and statistical modeling—ideally in an automated way that only requires them to precisely formulate a model and a question? Second: instead of building models by hand, we can often *learn* them from data. But how can we find the necessary structure in large datasets in a computationally tractable way? Third: building models is only the first step. Those models are often subsequently used to make decisions, such as what drugs to develop, what investments to make, or where to drill for oil. How can we make these decisions in mathematically principled ways (especially if one decision depends sequentially on another)? Fourth: how can we deal with rampant uncertainty—including missing data, noisy data, or our own subjective uncertainty about what kind of structure we think we need in the first place?

To answer these questions, I focus on the intersection of different parts of machine learning. I develop **probabilistic programming** languages that cleanly separate declarative knowledge of a problem from algorithms for solving that problem. I also work on the companion problem of **probabilistic inference**, and how we can best solve the problems we write down. I then study how to use these new languages to move beyond flat representations of structure to more complex representations that are naturally suited to modeling real-world situations—leveraging **structured prior knowledge** about a problem to interpret data in the context of **generative probabilistic models**. A particular emphasis of mine is **time series modeling**, and how to learn models of dynamical systems that support decision making, combined with **reinforcement learning** algorithms for making those decisions. I then use the tools I develop to solve applications such as inverse geological modeling, robot path planning, and multicore optimization.

1. Structured Probabilistic Modeling and Probabilistic Programming

I develop new ways of formulating and solving statistical inference problems, emphasizing general-purpose tools that are useful in real-world applications. Probabilistic programming is my main research focus.

Probabilistic Programming: Beyond Graphical Models

As probabilistic modeling becomes more vital to both industry and academia, it is becoming increasingly

important to provide tools to make it easy. Over the last 10 years, the language of graphical models has served as a bridge between statistics and machine learning, but it has limitations: graphical models lack the expressiveness to capture qualitative process knowledge, they lack the capacity for recursive and reflective reasoning, and they do not capture true model learning in a useful native form (no graphical model represents the fine-grained structure of the problem of learning the structure of a graphical model).

Probabilistic programming is a recent generalization of graphical models that addresses all these limitations. Rather than marry *statistics* with *graph theory*, probabilistic programming marries *Bayesian probability* with *computer science*: modelers specify a stochastic process using syntax that resembles a modern programming language, and allows them to define distributions using recursion, libraries, or data structures. And just as linear programming separates the specification of an optimization objective and problem constraints from algorithms that optimize the objective, probabilistic programs separate the declaration of a statistical model from algorithms that perform inference in that model.

My research pushes every aspect of these languages. I have developed novel implementation methods [5] that have served as the basis for new languages such as Bher (an implementation of the MIT-Church language) and pystoc (a new language based on python); I am also using the technique to develop *Stochastic Matlab*, an open-source probabilistic programming language with Matlab syntax and a variety of inference algorithms.

I also develop novel inference algorithms that blend concepts from statistics and other disciplines. For example, my recent work on nonstandard interpretations [3] marries programming language theory with statistics. I have also shown that variational inference can be considered a reinforcement learning problem [1], which suggests novel, effective techniques that leverage temporal structure within generative processes.

Scalable inference remains the primary challenge. My research will continue to focus on scalability through a mix of program analysis, compiler technology, advanced inference, and massive parallelism.

Structured Generative Models for Inverse Geology

Inverse geological modeling is a good example of how I apply probabilistic programming. Oil companies face a fundamental challenge: how can they tap process knowledge stored in the heads of their staff geologists, and bring it to bear on the problem of interpreting field data (such as seismic volumes or well logs)? In collaboration with Willsky and Shell Oil, I am developing probabilistic programs that model deepsea geological formations [1, 5]. Geologists provide forward models of deposition, erosion, and turbidite channel formation, which we analyze to build statistical models. Given seismic readings and well log data, the task is to infer underlying rock structure, which is then used in concert with oil flow models to guide decisions about where to drill for oil. The financial incentives are enormous: only one in four wells hit oil, and they can cost tens or hundreds of millions of dollars to drill. Our models and algorithms are state-of-the-art, but not specific to this application, implying they could be applied to many other geophysical and biological applications.

2. Dynamical Systems Modeling and Reinforcement Learning

I am deeply interested in dynamical systems modeling and the question of sequential decision making under uncertainty. This complements my research in probabilistic programming: I have used dynamical systems theory to improve statistical inference, and I also use probabilistic programming to model different kinds of dynamical systems and decision problems with rich structure.

Bayesian Reinforcement Learning

Reinforcement learning (RL) is where model building meets planning: agents must construct models of a world’s dynamics while simultaneously acting in it. I have contributed to the fundamentals of RL with research on efficient planning [15], massively parallel planners [23, 25, 27] (that can also accelerate the solution of large, sparse linear systems of equations [22]), as well as adaptive discretization for planning [26].

To move today’s RL algorithms towards more complex, real-world domains, they must be extended to leverage structure. I do this with *Bayesian RL*, which blends statistical models, learning and planning. For example, I have helped develop a Bayesian reinforcement learning algorithm which acts provably optimally with a distribution over possible world structures, and demonstrated that a nonparametric Bayesian model I contributed accelerates learning compared to less structured models [12].

But so far, researchers have only leveraged structure in world dynamics. Are there other opportunities to apply the Bayesian toolkit? In collaboration with Doshi-Velez, Kaelbling, Roy, Goodman and Tenenbaum, I have developed the idea of *policy priors* that capture structure in action spaces. This allows agents to leverage expert data in novel ways [9], search policy spaces more efficiently [4], and identify reusable grammar-like structure in policy space [2]. My research will continue to leverage advanced statistical modeling to find structure in value functions, reward functions, and possibly even algorithms themselves.

Dynamical Systems Modeling with Nonparametric Hierarchical Bayes

Time series problems include such diverse tasks as weather modeling, computational finance, and robot engineering. For these problems, can we build or learn a model which allows us to forecast the future, possibly using those forecasts to control the system optimally? Most time series models assume that the data is a simple sequence of scalar numbers, like temperatures or stock prices. But this is insufficient for many real-world time series problems. How can we move beyond simple representations, to situations where structure is best described using objects, events, physics, and intentional agents?

I use nonparametric Bayesian modeling to capture flexible structure in time series data. For example, I have developed the *Infinite Latent Events Model*, which combines ideas from hierarchical nonparametric Bayesian models, latent factor analysis and causal modeling to discover factored structure in temporal data [11]. Doshi-Velez and I have also developed the *Infinite Dynamic Bayes Net* [6] which simultaneously discovers factors, transitions, state representations and graphical structure in a dataset. How can we scale these approaches? How can we capture more structure? Computationally efficient, factored time series modeling would have significant impact in many fields, and is an interesting and challenging future direction I will pursue.

In previous work, I have also addressed the basic problem of determining a representation of *state* for a dynamical system. I have developed novel models, known as *Predictive State Representations* (PSRs) whose states are defined using only observable quantities in the future. This representation results in simple learning algorithms with attractive theoretical properties. For example, Rudary and I developed a PSR equivalent of the Kalman filter with a statistically consistent learning algorithm [24]—a powerful and uncommon learning guarantee. I subsequently generalized this from linear to nonlinear dynamics, resulting in state-of-the-art models for benchmark time series prediction tasks [20, 21]. This work led to an information-theoretic approach to optimizing state representations [18], as well as explorations of relational knowledge representations [19]. I have also unified a number of popular PSR models in a single general model [17], which was then used to predict and analyze new members of the class [14] and tested on additional applications [8, 16].

Smartlocks: Reinforcement Learning for Multicore Systems

The multicore revolution is imminent, but it is not easy to enable programmers to maximize the power of next generation systems. In lieu of ever more complicated compilers and ever more demanding code design, machine learning can be used to create adaptive algorithms which learn how to use computational resources. MIT's Carbon research group and I have developed the *Smartlock*, a mutex which adaptively arbitrates resources under contention [10, 13]. Who should get the lock next? The Smartlock uses RL to decide, thereby maximizing system throughput. We have applied Smartlocks to adapt to thermal throttling events in asymmetric multicores, and demonstrated that its adaptive policy is superior to any static policy. The Smartlock is generic enough to be used as a building block to construct intelligent variants of standard system components—replacing normal locks with Smartlocks in a scheduler could result in a smart scheduler, for instance. This could drive many future research projects to develop smart versions of power managers, thread schedulers, or disk controllers; we have started by developing smart data structures [7].

Future work in this area could have significant impact. We are faced with ever increasing complexity—not just in data, but also in systems. Improving the intersection of learning, uncertainty and decision making will enable us to engineer systems designed to be controlled by machine learning, opening many new applications.

3. Future Plans and Extensions

What further applications can we imagine if we had compelling solutions to fast model building and decision making? Such technology will have significant impact in many fields—perhaps enabling adaptive medical devices that build models of specific patients, or new computing systems with complexity beyond our ability to directly manage. But there are still large questions to be answered. What does the future of probabilistic modeling look like? How can it best be integrated with decision making? What are good languages for describing the intersection of the two? My goal is to answer these foundational questions. Breakthroughs in many fields—the rich hypothesis spaces of hierarchical Bayes; cheap computing power; massive datasets and advances in statistical learning theory—suggest that this is an exciting time to pursue such an agenda.