Safety, Risk Awareness and Exploration in Reinforcement Learning

Teodor Moldovan
Pieter Abbeel, Ed.
Michael Jordan, Ed.
Francesco Borrelli, Ed.

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2016-20
http://www.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-20.html

May 1, 2016

Safety, Risk Awareness and Exploration in Reinforcement Learning

by

Teodor Mihai Moldovan

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Pieter Abbeel, Chair
Professor Michael I. Jordan
Professor Francesco Borrelli

Spring 2014

Safety, Risk Awareness and Exploration in Reinforcement Learning

Abstract

Safety, Risk Awareness and Exploration in Reinforcement Learning

by

Teodor Mihai Moldovan

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Pieter Abbeel, Chair

Replicating the human ability to solve complex planning problems based on minimal prior knowledge has been extensively studied in the field of reinforcement learning. Algorithms for discrete or approximate models are supported by theoretical guarantees but the necessary assumptions are often constraining. We aim to extend these results in the direction of practical applicability to more realistic settings. Our contributions are restricted to three specific aspects of practical problems that we believe to be important when applying reinforcement learning techniques: risk awareness, safe exploration and data efficient exploration.

Risk awareness is important in planning situations where restarts are not available and performance depends on one-off returns rather than average returns. The expected return is no longer an appropriate objective because the law of large numbers does not apply. In Chapter 2 we propose a new optimization objective for risk-aware planning and show that it has desirable theoretical properties, relating it to previously proposed risk-aware objectives: minmax, exponential utility, percentile and mean minus variance. In environments with uncertain dynamics, exploration is often necessary to improve performance. Existing reinforcement learning algorithms provide theoretical exploration guarantees, but they tend to rely on the assumption that any state is eventually reachable from any other state by following a suitable policy. For most physical systems this assumption is impractical as the systems would break before any reasonable exploration has taken place. In Chapter 3 we address the need for a safe exploration method. In Chapter 4 we address the specific challenges presented by extending model-based reinforcement learning methods from discrete to continuous dynamical systems. System representations based on explicitly enumerated states are not longer applicable. To address this challenge we use a Dirichlet process mixture of linear models to represent dynamics. The proposed model strikes a good balance between compact representation and flexibility. To address the challenge of efficient exploration-exploitation trade-off we apply the principle of Optimism in the Face of Uncertainty that underlies numerous other provably efficient algorithms in simpler settings. Our algorithm reduces the exploration problem to a sequence of classical optimal control problems. Synthetic experiments illustrate the effectiveness of our methods.

To Ana, Doru and Alina

Contents

List of Figures **iv**

1 Introduction **1**

2 Chernoff Bounds for Markov Decision Processes **4**
2.1 Introduction 4
2.2 Background and Notation 6
2.2.1 The Markov Decision Process 6
2.2.2 Exponential Utility Optimization 7
2.3 The Chernoff Functional as Risk-Aware Objective 7
2.3.1 Introducing the Chernoff Functional 7
2.3.2 Properties 8
2.4 Optimizing the Proposed Objective 9
2.4.1 Problem Statement 9
2.4.2 Algorithm 9
2.4.3 Complexity Analysis 11
2.5 Experiments 12
2.5.1 Grid world 13
2.5.2 Air travel planning 15
2.6 Discussion 16

3 Safe Exploration for Markov Decision Processes **17**
3.1 Introduction 17
3.2 Notation and Assumptions 18
3.3 Problem formulation 19
3.3.1 Exploration Objective 19
3.3.2 Safety Constraint 20
3.3.3 Safety Counter-Examples 21
3.4 Guaranteed Safe, Potentially Sub-optimal Exploration 22
3.5 Experiments 25
3.5.1 Grid World 25

3.5.2 Martian Terrain . . . 27
3.5.3 Computation Time . . . 30
3.6 Discussion . . . 31

4 Exploration for Continuous Dynamical Systems 32
4.1 Introduction . . . 32
4.2 Bayesian Linear Model . . . 34
4.3 Dirichlet Process Mixture of Linear Models . . . 36
4.3.1 The Standard Dirichlet Process Mixture Model. . . . 37
4.3.2 Variational Inference. . . . 37
4.3.3 Dynamics Prediction. . . . 39
4.4 Optimal Control . . . 41
4.4.1 Continuous Time Formulation . . . 42
4.4.2 Non-Linear Program Formulation . . . 42
4.4.3 Solving the Non-Linear Program . . . 43
4.5 Optimism-Based Exploration . . . 44
4.6 Experiments . . . 46
4.6.1 Under-Actuated Pendulum Swing-Up and Balance. . . . 47
4.6.2 Cartpole Swing-Up and Balance. . . . 49
4.6.3 Double Pendulum Swing-Up and Balance. . . . 50
4.6.4 Helicopter Learning Inverted Hover. . . . 51
4.7 Discussion . . . 54

5 Summary of Contributions 55

Bibliography 57

A Additional Proofs and Examples 63
A.1 Example MDP where optimizing cost percentiles is counter-intuitive . . . 63
A.2 Proof of Theorem 1 . . . 63
A.3 Proof of Theorem 2 . . . 66
A.4 Proof of Theorem 3. . . . 66
A.5 Proof of Theorem 4. . . . 67

List of Figures

2.1 Qualitative example of the function f defined in Equation (2.2) and the approximation $\hat{f}$ computed by Algorithm 1. 10
2.2 Chernoff policies for the Grid World MDP. See text for complete description. The colored arrows indicate the most likely paths under Chernoff policies for different values of δ. The minimaxpolicy ($\delta = 0$) acts randomly since it always assumes the worst outcome, namely that any action will lead to a trap. . . . 13
2.3 Chernoff policies to travel from Shreveport Regional Airport (`SHV`) to Rafael Hernández Airport (`BQN`) at different risk levels. We are using International Air Transport Association (IATA) airport codes. 14
2.4 Histograms demonstrating the efficiency and relevance of our algorithm on 500 randomly chosen origin - destination airport pairs, at 15 risk levels. . . . 16

3.1 Starting from state S, the policy (aababab...) is safe at a safety level of .8. However, the policy (acccc...) is not safe since it will end up in the sink state E with probability 1. State-action Sa and state B can neither be considered safe nor unsafe, since both policies use them. 21
3.2 Under out belief the two MDPs above both have probability .5. It is intuitively unsafe to go from the start state S to B since we wouldn't know whether the way back is via U or L, even though we know for sure that a return policy exists. 21
3.3 The two MDPs on the left both have probability .5. Under this belief, starting from state A, policy (aaa...) is unsafe. However, under the mean transition measure, represented by the MDP on the right, the policy is safe. 22
3.4 Exploration experiments in simple grid worlds. See text for full details. Square sizes are proportional to corresponding state heights between 1 and 5. The large, violet squares have a height of 5, while the small, blue squares have a height of 1. Gray spaces represent states that have not yet been observed. Each row corresponds to the same grid world. The first column shows the belief after the first exploration step, while the second and third columns show the entire trajectory followed by different explorers. 26

3.5 Exploration efficiency comparison. We are showing the median, the upper and the lower quartiles of the fraction of the grid world that was uncovered by different explorers in randomly generated grid worlds. The "amount" of non-ergodicity is controlled by randomly making a fraction of the squares inaccessible (walls). We ran 1000, 500, 100,20 experiments for grids of sizes 10^2, 20^2, 30^2 and 40^2 respectively. We are comparing against our own adapted R-MAX exploration objective, the original R-MAX objective [22] and the Near-Bayesian exploration objective [43]. The last two behave identically in our grid world environment, since, once a state is visited, all transitions out of that state are precisely revealed. 28
3.6 Simulated safe exploration on a 2km by 1km area of Mars at -30.6 degree latitude and 202.2 degrees longitude, for 15000 time steps, at different safety levels. See text for full details. The color saturation is inversely proportional to the standard deviation of the height map under the posterior belief. Full coloration represents a standard deviation of 1cm or less. We report the difference between the entropies of the height model under the prior and the posterior beliefs as a measure of performance. Images: NASA/JPL/University of Arizona. 29

4.1 Modeling a scalar response Y of a scalar covariate X jointly as a two-dimensional Normal-Inverse-Wishart distribution may be used for Bayesian linear regression via the conditional distribution of Y given X. The blue ellipse show a level set of posterior Normal-Inverse-Wishart distribution. The red line show a "slice" at a given X corresponding to the conditional distribution of Y given X. 35
4.2 Clusters learned by batch variational inference on the trajectories of two dynamical systems under random controls. Both systems are described in Section 4.6. Extra dimensions corresponding to accelerations and control values are not shown. The different cluster patters reflect the different structures of non-linearities. 38
4.3 Modeling a scalar response Y of a scalar covariate X jointly as a mixture model may be used for Bayesian linear regression via the conditional distribution of Y given X. The blue ellipses show level sets of the Normal-Inverse-Wishart clusters. The red line show a "slice" at a given X corresponding to the conditional distribution of Y given X. The dotted red line shows the closest Normal-Inverse-Wishart approximation in KL divergence. 40
4.4 Pendulum swing-up and balance experiments. The goal is to swing the pendulum up from the initial state hanging down to the upright balanced state. One swing is necessary to provide enough momentum due to torque constraints. 48
4.5 Cartpole swing-up and balance experiments. The goal is to swing the pendulum up from the initial state hanging down to the upright balanced state. . . 49

4.6 Double pendulum swing-up and balance experiments. The goal is to swing both pendulums up from the initial state hanging down to the upright balanced state. Torque constraints require building up energy by swinging. . . 51
4.7 Helicopter learning inverted hover. The initial state (left) corresponds to upright hover but unknown dynamics. The goal state (right) is inverted hover. 52

A.1 Example showing that percentile optimization for MDPs is sometimes counter-intuitive. We are optimizing $F_{0.1}^{-1}[V(\pi)]$. Changing parameters of state S_2 affects the policy in state S_3 even though S_2 is unreachable from S_3. 64
A.2 MDP reduction of the 3SAT problem. 67

Chapter 1

Introduction

Humans display an impressive level of competence when learning to control dynamical systems; we naturally account for stochastic disturbances, we make sure to never stray too far from "safety", we learn to make approximate predictions from experience and we choose informative exploratory actions. As an example consider a novice pilot learning to fly a remote controlled helicopter. In the early stages of learning we might see the main rotor spinning up until the helicopter is almost ready to take off, then spinning back down; the pilot has already learned that the most relevant control is the collective and is exploring its effects. Keeping the helicopter close to the ground is a "safety" feature meant to prevent large deviations from the horizontal orientation and, thus, ensure that future trials are possible without crashing. Additionally the novice pilot might also manually reposition the helicopter so that all trials start roughly from the same state usually situated in the center of the room. Doing so ensures that past observations are relevant to the new trial and minimizes the risk that random disturbances will bring the helicopter dangerously close to walls. When learning more complex maneuvers pilots usually chose an appropriate "home state", typically corresponding to hover in the center of the room. From this home state they execute exploratory trajectories aiming to get successively closer to a "goal state" such as inverted hover, while making sure that they can recover from unexpected outcomes and return to the home state relying only on currently developed skills. Most impressively, pilots have the ability to (re)discover aerobatic maneuvers without the need of expert demonstrations, relying solely on trial and error.

In an effort to bring reinforcement learning closer to this type of real-world learning ability we single out and investigate three features that we consider important: risk-aware planning, safety when exploring and efficient exploration for continuous dynamical systems.

Although sensing is critically important to the learning process in practice, we do not address it here and we assume that (noisy) observations of state are directly available. Similarly, we acknowledge that experiments on physical hardware present numerous significant challenges and we restrict our investigation to simulated domains. We propose general principles for safety when exploring and risk-awareness but only show algorithms for discrete

space planning problems. Our main focus is formulating planning problems that correspond to the desired behavior; the methods used to solve these planning problems are outside the scope of this work and we use off-the-shelf planners whenever possible.

Risk aware planning is the problem of choosing a sequences of actions to obtain reasonably good rewards while safeguarding against improbable large loses. Chapter 2 is devoted to this topic and follows the previously published article by Moldovan and Abbeel [52] with minor extensions. Risk aware planning is preferable in situations where the environment does not provide the option of consequence-free restarts, so the law of large numbers does not apply. By contrast, in situations where the law of large numbers does apply (when independent and identically distributed versions of the planning scenario are accessible) optimizing expected rewards is the norm since the distribution of the average reward converges to the expectation. Risk-aware planning is relevant in most practical situations because restarts are usually costly or not available. The main difficulty is the absence of a generality accepted formal definition of the intuitive notion of "risk-awareness". We propose a new risk-aware objective based on optimal Chernoff bounds and theoretically show its generality; specifically we show that it subsumes a number of other previously proposed risk-aware objectives: minimax, exponential utility, percentile minimax and, in the limit, expected return. Additionally we show a provably efficient algorithm for planning with the newly proposed objective based on sequential exponential utility optimization. We demonstrate our approach on a number of synthetic grid worlds that are easy to analyse and on a problem based on real-world data: planning air travel considering the risk of delayed flights.

Safety in the context of exploration refers to the trade-off between taking actions that are more informative (flying the helicopter fast) and ensuring the prospect of more exploration in the future. Chapter 3 is devoted to this topic and follows the previously published article by Moldovan and Abbeel [53] with minor extensions. Safety differs from risk-awareness in that no specific rewards are involved. Safety is important in situations where overly aggressive control without the ability to make accurate predictions would lead to crashes requiring costly repairs before exploration may resume. Unfortunately the theoretical guarantees of most previously proposed exploration algorithms are restricted to domains where safe return to a start state is guaranteed for all actions (*ergodic* domains). The ergodic assumption allows for exploration algorithms that operate by simply favoring states that have rarely been visited before. For most physical systems this assumption is impractical as the systems would break before any reasonable exploration has taken place, i.e., most physical systems don't satisfy the ergodicity assumption. We propose a novel and general formulation of safety through ergodicity. According to our definition, safe policies are those that guarantee return to the start state with high probability even when the outcome of actions taken is uncertain. We show that imposing safety by restricting attention to the resulting set of guaranteed safe policies is NP-hard and subsequently proceed to presenting an efficient algorithm for guaranteed safe, but potentially suboptimal, exploration. At its core is an optimization formulation in which the constraints restrict attention to a subset of the guaranteed safe policies and the objective favors exploration policies. Our framework is compatible with the majority of

previously proposed exploration methods, which rely on an exploration bonus. Our experiments, which include a Martian terrain exploration problem, show that our method is able to explore better than classical exploration methods in non-ergodic environments.

Exploration for continuous dynamical systems is the problem of choosing exploratory actions when system dynamics are uncertain. Chapter 4 is devoted to this topic and substantially extends the results presented by Moldovan, Jordan and Abbeel at the Reinforcement Learning and Decision Making Meetings (RLDM) 2013. The goal when exploring is to reduce uncertainty in the dynamics as much as necessary to reach a pre-defined target state. Exploration is important when system dynamics change in the course of operation (for example because of wear or damage), when each individual system is unique due to variations in manufacturing or when the environment changes (for example traversing different types of terrains). Although numerous theoretically justified methods address the problem of efficient exploration for discrete state spaces, most of these methods do not naturally extend to continuous dynamical systems. We address this challenge with a newly proposed exploration algorithm based on the principle of *Optimism in the Face of Uncertainty*, the same principle that motivates a large number of successful exploration algorithms in simpler settings. To represent the uncertain system dynamics in a Bayesian framework we use the Dirichlet process mixture of linear models; its advantages are discussed. Experiments show that our method can solve swing-up and balance tasks for cartpole, under-actuated pendulum and double pendulum as well as helicopter inverted hover, all starting with no knowledge of the dynamics. The approach is highly automated requiring minimal task-specific parameter tuning.

Chapter 2

Chernoff Bounds for Markov Decision Processes

2.1 Introduction

The *expected* return is often the objective function of choice in planning problems where outcomes not only depend on the actor's decisions but also on random events. Often expectations are the natural choice, as the law of large numbers guarantees that the average return over many independent runs will converge to the expectation. Moreover, the linearity of expectations can often be leveraged to obtain efficient algorithms.

Some games, however, can only be played once, either because they take a very long time (investing for retirement), because we are not given a chance to try again if we lose (skydiving, crossing the road), or because i.i.d.versions of the game are not available (stock market). In this setting, we can no longer take advantage of the law of large numbers to ensure that the return is close to its expectation with high probability, so the expected return might not be the best objective to optimize. If we were pessimistic, we might assume that everything that can go wrong will go wrong and try to minimize the losses under this assumption. This is called minmax optimization and is sometimes useful, but, most often, the resulting policies are overly cautious. A more balanced and general approach would include minmax optimization and expectation optimization, corresponding respectively to absolute risk aversion and risk ignorance, but would also allow a spectrum of policies between these extremes.

As a motivating example, consider buying tickets to fly to a very important meeting. Shorter travel time is preferable, but even more importantly, it would be disastrous if you arrived late. Some flights arrive on time more often than others, and the delays might be amplified if you miss connecting flights. With these risks in mind, would you rather take a route with an expected travel time of 12:21 and no further guarantees, or would you prefer a route that takes less than 16:19 with 99% probability? Our method produces these options

when traveling from Shreveport Regional Airport (SHV) to Rafael Hernández Airport (BQN). According to historical flight data, if you chose the former alternative you could end up travelling for 22 hours with 8% probability. Another example comes from software quality assurance. Amazon.com requires its sub-services to report and optimize performance at the 99.9^{th} percentile, rather than in expectation, to make sure that all of its customers have a good experience, not just the majority [26]. In the economics literature, this percentile criterion is known as *value at risk* and has become a widely used measure of risk after the market crash of 1987 [40]. At the same time, the classical method for managing risk in investment is Markovitz portfolio optimization where the objective is to optimize expectation minus weighted variance. These examples suggest that proper risk-aware planning should allow a trade-off between expectation and variance, and, at the same time, should provide some guarantees about the probability of failure.

Risk-aware planning for Markov decision processes (MDPs) is difficult for two main reasons. First, optimizing many of the intuitive risk-aware objectives seems to be intractable computationally. Both mean minus variance optimization and percentile optimization for MDPs have been shown to be NP-hard in general [50, 31]. Consequently, we can only optimize relaxations of these objectives in practice. Second, it seems to be difficult to find an optimization objective which correctly models our intuition of risk awareness. Even though expectation, variance and percentile levels relate to risk awareness, optimizing them directly can lead to counterintuitive policies as illustrated recently in [50], for the case of mean minus variance optimization, and in the appendix of this thesis, for percentile optimization.

Planning under uncertainty in MDPs is an old topic that has been addressed by many authors. The minmax objective has been proposed by [59, 38], who propose a dynamic programming algorithm for optimizing it efficiently. Unfortunately, minmax policies tend to be overly cautious. A number of methods have been proposed for relaxations of mean minus variance optimization [50, 46]. Percentile optimization has been shown to be tractable when dealing with ambiguity in MDP parameters [32, 55], and it has also been discussed in the context of risk [20, 65]. Our approach is closest to the line of work on exponential utility optimization [51, 19]. This problem can be solved efficiently and the resulting policies conform to our intuition of risk awareness. However, previous methods give no guarantees about probability of failure or variance. For an overview of previously used objectives for risk-aware planning in MDPs, see [27, 45].

Our method arises from approaching the problem in the context of probability theory. We observe connections between exponential utility maximization, Chernoff bounds, and cumulant generating functions, which enables formulating a new optimization objective for risk-aware planning. This new objective is essentially a re-parametrization of exponential utility, and inherits both the efficient optimization algorithms and the concordance to intuition about risk awareness. We show that optimizing the proposed objective includes, as limiting cases, both minmax and expectation optimization and allows interpolation between them. Additionally, we provide guarantees at a certain percentile level, and show connections to mean minus variance optimization.

Two experiments, one synthetic and one based on real-world data, support our theoretical guarantees and showcase the proposed optimization algorithms. Our largest MDP has 124791 state-action pairs—significantly larger than experiments in most past work on risk-aware planning. Our experiments illustrate the ability of our approach to—out of the exponentially many policies available—produce a family of policies that agrees with the human intuition of varying risk.

This Chapter follows the previously published article by Moldovan and Abbeel [52] with minor extensions.

2.2 Background and Notation

2.2.1 The Markov Decision Process

An MDP consists of a state space $\mathcal{S}$, an action space $\mathcal{A}$, state transition dynamics, and a cost function G. Assume that, at time t, the system is in state $s_t \in \mathcal{S}$. Once the player chooses an action $a_t \in \mathcal{A}$, the system transitions stochastically to state $s_{t+1} \in \mathcal{S}$, with probability $p(s_{t+1}|s_t, a_t)$, and the player incurs a stochastic cost of $G^t(s_t, a_t, s_{t+1})$. The process continues for a number of time steps, h, called the *horizon*. We eventually care about the total cost obtained. We represent the player's strategy as a time dependent *policy*, which is a measure on the space of state-actions. Finally, we set the starting state to some fixed $s_0 \in \mathcal{S}$. Then, the objective is to "optimize" the *random variable* J^h, defined by $J^h := \sum_{t=0}^{h-1} G^t(S_t, A_t, S_{t+1})$.

Traditionally, "optimizing" J means minimizing its expected value (solving $\min_\pi E_{s,\pi}[J]$). The classical solution to this problem is to run *value iteration*, summarized below:

$$q^{t+1}(s,a) := \sum_{s'} p_{s'|s,a} \left(G^t_{s,a,s'} + j^t(s') \right)$$

$$j^t(s) := \min_a q^t(s,a) = \min_\pi E_{s,\pi}[J^t], \qquad \pi^t(s,a) := 1\{q^t(s,a) = j^t(s)\}$$

We will refer to policies obtained by standard value iteration as *expectimin* policies. We use upper case letters for random variables. We assume that the state-action space is finite and that sums with zero terms, for example J^0, are equal to zero. The notation $E_{s,\pi}$ signifies taking the expectation starting from $S_0 = s$, and following policy π.

We assume that costs are upper bounded, that is there exists j_M such that $J \leq j_M$ almost surely for any start state and any policy, and that the expected costs are finite. Finally, in this Chapter we will not consider discounting explicitly. If necessary, discounting can be introduced in one of two ways: either by adding a transition from every state, for all actions, to an absorbing "end game" state, with probability γ, or by setting a time dependent cost as $G^t_{\text{new}} = \gamma^t G^t_{\text{old}}$. Note that these two ways of introducing discounting are equivalent when optimizing the expected cost, but they can differ in the risk-aware setting we are considering. We refer the reader to [62] and [15] for further background on MDPs.

2.2.2 Exponential Utility Optimization

An alternative way to "minimize" the random cost J is *exponential utility optimization*, that is solving $\min_\pi E_{s,\pi}\left[e^{zJ}\right]$ for a fixed value of the parameter $z > 0$. The resulting policies are known to be more or less risk aware depending on the value of the parameter [51, 19]. Fortunately, exponential utility optimization for MDPs can be solved as efficiently as expected cost minimization by an iterative algorithm similar to value iteration, which we call *cumulant generating function iteration (CGF iteration)*:

$$u^{t+1}(s,a) := \log \sum_{s'} p_{s'|s,a} \exp\left(\log E\left[e^{zG^t_{s,a,s'}}\right] + w^t(s')\right)$$
$$w^t(s) := \min_a u^t(s,a) = \min_\pi \log E_{s,\pi}[e^{zJ^t}], \qquad \pi^t(s,a) := 1\{u^t(s,a) = w^t(s)\},$$

where we assume that the random step-wise costs, $G^t_{s,a,s'}$, have cumulant generating functions, which is the case for bounded costs, normally distributed costs, and many others. The algorithm we propose in this thesis repeatedly uses CGF iteration as a subroutine to compute the optimal exponential utility, and the corresponding optimal policy, for different values of the parameter.

2.3 The Chernoff Functional as Risk-Aware Objective

2.3.1 Introducing the Chernoff Functional

We propose optimizing the following functional of the cost, which we call the *Chernoff functional* since if often appears in proving Chernoff bounds:

$$C^\delta_{s,\pi}[J] = \inf_{\theta>0}\left(\theta \log E_{s,\pi}\left[e^{J/\theta}\right] - \theta\log(\delta)\right). \tag{2.1}$$

First, note the total cost appears in the expression of the Chernoff functional as an exponential utility ($E_{s,\pi}[e^{J/\theta}]$). This shows that there is a strong connection between our method and exponential utility optimization. Specifically, all polices proposed by our algorithm, including the final solution, are optimal policies with respect to the exponential utility for some parameter. These policies are known to show risk-awareness in practice [51, 19], and our method inherits this property. In some sense, our proposed objective is *re-parametrization* of exponential utility, which was obtained through its connections to Chernoff bounds and cumulant generating functions.

The theorem below, which is one of the main contributions of this Chapter, provides more reasons for optimizing the Chernoff functional in the risk-aware setting. We will state and discuss the theorem here, but leave the proof for the appendix.

2.3.2 Properties

Theorem 1. *Let $\delta \in [0, 1]$, and let J be a random variable that is upper bounded. Then, the Chernoff functional of this random variable, $C^\delta[J]$, is well defined, and has the following properties:*

(i) $P(J \geq C^\delta[J]) \leq \delta$

(ii) $C^1[J] = \lim_{\theta\to\infty} \theta \log E[e^{J/\theta}] = E[J]$

(iii) $C^0[J] := \lim_{\delta\to 0} C^\delta[J] = \lim_{\theta\to 0} \theta \log E[e^{J/\theta}] = \sup\{j : P\{J \geq j\} > 0\} < \infty.$

(iv) $C^\delta[J] = E[J] + \sqrt{2\log(1/\delta)\, Var[J]}$ *if J is Gaussian.*

(v) *As* $\delta \to 1$, $C^\delta[J] \approx E[J] + \sqrt{2\log(1/\delta)\, Var[J]}$

(vi) $C^\delta[J]$ *is a smooth, decreasing function of* δ.

Proof sketch. Property (i) is simply a Chernoff bound and follows by applying Markov's inequality to the random variable $e^{J/\theta}$. Property (iv) follows from the fact that all but the first two cumulants of Gaussian random variables are zero [42]. Properties (ii), (iii), (v) and (vi) follow from the following properties of *cumulant generating function*, $\log Ee^{zJ}$, [42]:

(a) $\log Ee^{zJ} = \sum_{i=1}^{\infty} z^i k_i/i!$ where k_i are the *cumulants* [42]. The coefficients k_i in general don't have simple expressions, but for $i = 1, 2$ we have $k_1 = E[J]$, $k_2 = \mathrm{Var}[J]$.

(b) $\log Ee^{zJ}$ as a function of $z \in \mathbb{R}$ is strictly convex, analytic and infinitely differentiable in a neighborhood of zero, if it is finite in that neighborhood.

□

Properties (ii) and (iii) show that we can use the δ parameter to interpolate between the nominal policy, which ignores risk, at $\delta = 1$, and the minmax policy, which corresponds to extreme risk aversion, at $\delta = 1$. Property (i) shows that the value of the Chernoff functional is with probability at least $1 - \delta$ an upper bound on the cost obtained by following the corresponding Chernoff policy. These two observations suggests that by tuning δ from 0 to 1 we can find a family of risk-aware policies, in order of risk aversion. Our experiments support this hypothesis (Section 2.5).

Property (i) shows a connection between our approach and percentile optimization. Although we are not optimizing the δ-percentile directly, our method provides guarantees about it. Properties (v) and (iv) show a connection between optimizing the Chernoff functional and mean minus variance optimization, which has been proposed before for risk-aware planning, but was found to be intractable in general [50]. Via property (v), we can optimize mean minus variance with a low weight on variance if we set δ close to 1. In the limit, this allows us to optimize the expectation, while breaking ties in favor of lower variance.

Property (iv) show that we can optimize mean minus scaled standard deviation exactly if the total cost is Gaussian. Typically, this will not be the case, but, if the MDP is ergodic and the time horizon is large enough, the total cost will be close to Gaussian, by the central limit theorem. The see why this is true, note that, by the Markov property, costs between successive returns to the same state are i.i.d.random variables [33]. Our formulation ties into mean minus standard deviation optimization, which is of consistent dimensionality, unlike the classical mean minus variance objective.

2.4 Optimizing the Proposed Objective

2.4.1 Problem Statement

Finding the policy that optimizes our proposed objective at a given risk level δ amounts to a double optimization problem:

$$\begin{aligned} \min_{\pi} C^{\delta}_{s,\pi}[J] &= \inf_{\theta>0} \left(\theta \cdot \min_{\pi} \log E_{s,\pi}\left[e^{J/\theta}\right] - \theta \log(\delta) \right) \\ &= \inf_{\theta>0} \left(f(\theta) - \theta \log(\delta) \right) \\ \text{where} \quad f(\theta) &:= \theta \cdot \min_{\pi} \log E_{s,\pi}\left[e^{J/\theta}\right] \end{aligned} \tag{2.2}$$

The inner optimization problem, the optimization over policies π, is simply exponential utility optimization, a classical problem that can be solved efficiently as we discussed in Section 2.2.2. Figure 2.1 shows a qualitative plot of the function defined above. The main difficulty is solving the outer optimization problem, over the scale variable θ. Unfortunately, this problem is not convex and may have a large number of local minima. Our main algorithmic contribution consists of an approach for solving the outer (non-convex) optimization problem efficiently to some specified precision ε.

Note that the Bellman optimality principle does not hold for our objective. In general, the policy that optimizes the Chernoff functional for some starting state is not the same as the policy that optimizes the Chernoff functional starting from the subsequent state. Formally, it is not the case that $\operatorname{argmax}_{\pi} C^{\delta}_{s,\pi}[J^h] = \operatorname{argmax}_{\pi} C^{\delta}_{S',\pi}[J^{h-1}]$, where S' is the successor state of s. For the principle to hold, one would have to allocate part of the risk to what happens before time t and part of the risk to what happens after time t. This requires either pre-allocation of risk, or incorporation of all possible risk scenarios into the state (which leads to an impractical increase in state-space size for most problems). Our approach automatically allocates risk over the entire duration of the MDP.

2.4.2 Algorithm

Based on Theorems 1 and 2 (below), we propose a method for finding the policy that minimizes the Chernoff functional, to precision ε, with worst case time complexity $O(h|S|^2|A|/\varepsilon)$.

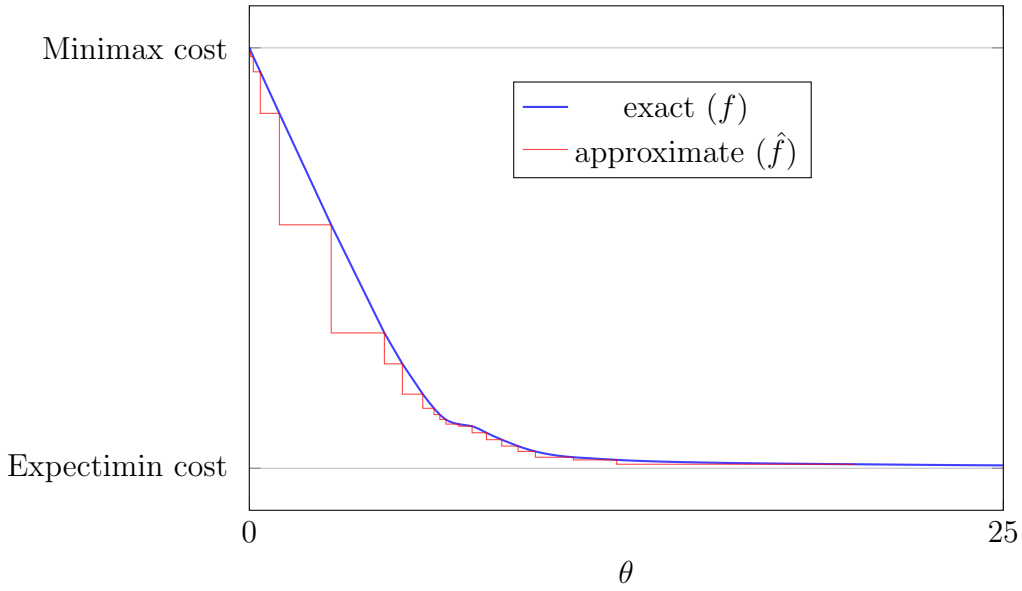


Figure 2.1: Qualitative example of the function f defined in Equation (2.2) and the approximation $\hat{f}$ computed by Algorithm 1.

It is summarized in Algorithm 1. Our approach is solving the optimization problem in (2.2) with an approximation of the function f. The algorithm maintains such an approximation and improves it as needed up to a precision of ε. Importantly, the same approximate function is shared between subsequent optimization problems (for different values of δ), explaining the somewhat surprising result that the worst case time complexity (expressed in terms of number evaluations of f) does not have a strong dependence on the number of δ values.

Properties (ii) and (iii) of Theorem 1 imply that $f(0)$ can be computed by minimaxoptimization and $f(\infty)$ can be computed by value iteration (expectimin optimization), which both have the same time complexity as exponential utility optimization: $O(h|S|^2|A|)$. Once we have computed these limits, the next step in the algorithm is finding some appropriate bounding interval, $[\theta_1, \theta_2]$, such that $f(0) - f(\theta_1) < \varepsilon$ and $f(\theta_2) - f(\infty) < \varepsilon$. We do this by first searching over $\theta = 1, .1, 10^{-2}, \cdots$, and, then, over $\theta = 1, 10, 10^2, \cdots$. For a given machine architecture, the number of θ values is bounded by the number format used in the implementation. For example, working with double precision floating-point numbers limits the number of θ evaluations to $2 \cdot 1023$, implied by the fact that exponents are only assigned 11 bits. In our experiments, this step takes 10-15 function evaluations.

Now, for any given risk level, δ, we will find θ^* that minimizes the objective, $f(\theta) - \theta \log(\delta)$, among those θ where we have already evaluated f. We will, then, evaluate f at a new point: the geometric mean of θ^* and its closest neighbor to the right. We stop iterating when the function value at the new point is less than ε away from the function value at θ^*, and return the corresponding optimal exponential utility policy. Consequently, our algorithm evaluates

Algorithm 1 Near optimal Chernoff bound algorithm

Require: Markov Decision Process, MDP.
Require: error level, $\varepsilon > 0$, safety level δ.
  $f_m \leftarrow M$.expectimin value(), $f_M \leftarrow M$.minmax value()
  $\hat{f} \leftarrow$ new empty hash map
  $\hat{f}[1] \leftarrow M$.optimal exponential utility(1)
  repeat
    $\theta_1 \leftarrow \min \hat{f}$.keys()/10 ($\hat{f}$.keys() is the set of keys stored in the hash map $\hat{f}$)
    $\hat{f}[\theta_1] \leftarrow \theta_1 \cdot M$.optimal exponential utility($1/\theta_1$)
  until $f_M - \hat{f}[\theta_1] < \varepsilon$
  repeat
    $\theta_2 \leftarrow \max \hat{f}$.keys() $\cdot\, 10$
    $\hat{f}[\theta_2] \leftarrow \theta_2 \cdot M$.optimal exponential utility($1/\theta_2$)
  until $\hat{f}[\theta_2] - f_m < \varepsilon$
  repeat
    $\theta^* \leftarrow \text{argmin}_{\theta<\theta_2, \theta\in\hat{f}.\text{keys}()}(\hat{f}[\theta] - \theta \log(\delta))$,
    $\theta_n \leftarrow \min\{\theta > \theta^*, \theta \in \hat{f}.\text{keys}()\}$
    $\theta_c \leftarrow \sqrt{\theta^* \cdot \theta_n}$
    $\hat{f}[\theta_c] \leftarrow \theta_c \cdot M$.optimal exponential utility($1/\theta_c$)
  until $\hat{f}[\theta^*] - \hat{f}[\theta_c] < \varepsilon$
  $\pi \leftarrow M$.optimal exponential utility policy($1/\theta^*$)
  return π

f at a subset of the points $\{\theta_1(\theta_2/\theta_1)^{i/n} : i = 0, \cdots, n\}$ where n is a power of 2. Theorem 2 guarantees that to get an ε guarantee for the accuracy of the optimization it suffices to perform $n(\varepsilon) = O(1/\varepsilon)$ evaluations of f, where we are now treating $\log(\delta_2) - \log(\delta_1)$ as a constant. Therefore, the number of functions evaluations is $O(1/\varepsilon)$, and, since the time complexity of every evaluation is $O(h|S|^2|A|)$, the total time complexity of the algorithm is $O(h|S|^2|A|/\varepsilon)$.

2.4.3 Complexity Analysis

Theorem 2. *Consider the interval $0 < \theta_1 < \theta_2$ split up into n sub-intervals by $\theta_i^n = \theta_1(\theta_2/\theta_1)^{i/n}$, and let $\hat{f}_n(\theta) := f(\max_{i\in 0\cdots n}\{\theta_i^n < \theta\})$ be our piecewise constant approximation to the function $f(\theta)$ defined in Equation (2.2). Then, for a given approximation error ε there exists $n(\varepsilon) = O((\log(\delta_2) - \log(\delta_1))/\varepsilon)$ such that $|\hat{f}_{n(\varepsilon)}(\theta) - f(\theta)| \leq \varepsilon$ for all $\theta \in [\theta_1, \theta_2]$.*

Proof sketch. The key insight when proving this theorem is bounding rate of change of f. We can immediately see that $f_\pi(\theta) := \theta \log E_{s,\pi}\left[e^{J/\theta}\right]$ is a convex function since it is the perspective transformation of a convex function, namely, the cumulant generating function

of the total cost J. Additionally, Theorem 1 shows that f_π is lower bounded by $E_{s,\pi}[J]$, assumed to be finite, which implies that f_π is non-increasing. On the other hand, by directly differentiating the definition of f_π, we get that $\theta f'_\pi(\theta) = f_\pi(\theta) - E_{s,\pi}[Je^{J/\theta}]/E_{s,\pi}[e^{J/\theta}]$.

Since we assumed that the costs, J, are upper bounded, there exist a maximum cost j_M such that $J \leq j_M$ almost surely for any starting state s, and any policy π. We have also shown that $f_\pi(\theta) \geq E_{s,\pi}[J] \geq j_m := \min_{\pi'} E_{s,\pi'}[J]$, so we conclude that, for any policy, π, the following holds:

$$-(j_M - j_m)/\theta \leq f'_\pi(\theta) \leq 0. \tag{2.3}$$

Now that we have bounded the derivative of f_π we can see that the value of f can not change too much over an interval $[\theta^n_{i+1}, \theta^n_i]$. Let $\pi_i := \operatorname{argmin}_\pi f_\pi(\theta^n_i)$ and $\pi_{i+1} := \operatorname{argmin}_\pi f_\pi(\theta^n_{i+1})$. Then:

$$\begin{aligned} 0 \leq f(\theta^n_i) - f(\theta^n_{i+1}) &= f_{\pi_i}(\theta^n_i) - f_{\pi_{i+1}}(\theta^n_{i+1}) \leq f_{\pi_{i+1}}(\theta^n_i) - f_{\pi_{i+1}}(\theta^n_{i+1}) \leq \\ &\leq \max_{\theta^n_i \leq \theta \leq \theta^n_{i+1}} |f'_{\pi_{i+1}}(\theta)| \cdot (\theta^n_{i+1} - \theta^n_i) = -f'_{\pi_{i+1}}(\theta^n_i) \cdot (\theta^n_{i+1} - \theta^n_i) \leq \\ &\leq (j_M - j_m) \cdot \frac{\theta^n_{i+1} - \theta^n_i}{\theta^n_i} = (j_M - j_m)\left(\left(\frac{\theta_2}{\theta_1}\right)^{1/n} - 1\right), \end{aligned} \tag{2.4}$$

where we first used the fact that $f_{\pi_i}(\theta^n_i) = \min_\pi f_\pi(\theta^n_i) \leq f_{\pi_{i+1}}(\theta^n_i)$, then the convexity of $f_{\pi_{i+1}}$ which implies that $f'_{\pi_{i+1}}$ is increasing, and, finally, our previous result in Equation (2.3). Our final goal is to find a value of $n(\varepsilon)$ such that the last expression in Equation (2.4) is less than ε. One can easily verify that the following $n(\varepsilon)$ satisfies this requirement (the detailed derivation appears in the Appendix):

$$n(\varepsilon) = \left\lceil \frac{(j_M - j_m)}{\varepsilon} \log\left(\frac{\theta_2}{\theta_1}\right) + \log\left(\frac{\theta_2}{\theta_1}\right) \right\rceil.$$

□

2.5 Experiments

We ran a number of experiments to test that our proposed objective indeed captures the intuitive meaning of risk-aware planning. The first experiment models a situation where it is immediately obvious what the family of risk-aware policies should be. We show that optimizing the Chernoff functional with increasing values of δ produces the intuitively correct family of policies. The second experiment shows that our method can be applied successfully to a large scale, real world problem, where it is difficult to immediately "see" the risk-aware family of policies.

Our experiments empirically confirm some of the properties of the Chernoff functional proven in Theorem 1: the probability that the return is lower than the value of the Chernoff

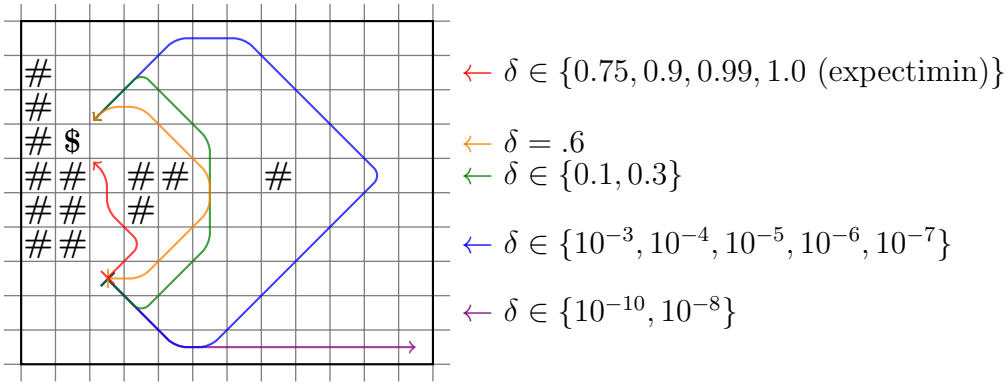
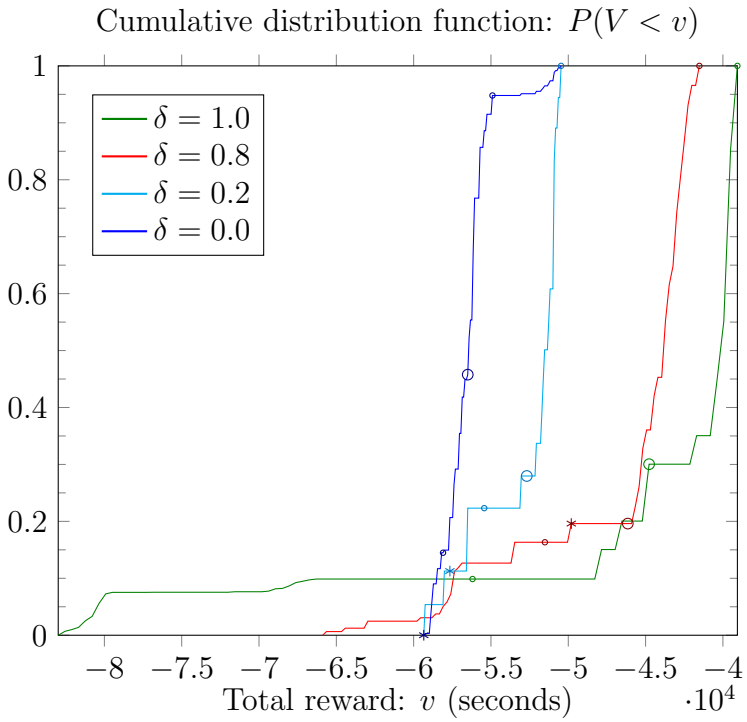


Figure 2.2: Chernoff policies for the Grid World MDP. See text for complete description. The colored arrows indicate the most likely paths under Chernoff policies for different values of δ. The minimaxpolicy ($\delta = 0$) acts randomly since it always assumes the worst outcome, namely that any action will lead to a trap.

policy at level δ is always less than δ, setting $\delta = 1$ corresponds to optimizing the expected return with the added benefit of breaking ties in favor of lower variance, and setting $\delta = 0$ leads to the minmax policy whenever it is defined. Additionally, we observed that policies at lower risk levels, δ, tend to have lower expectation but also lower variance, if the structure of the problem allows it. Generally, the probability of extremely bad outcomes decreases as we lower δ.

2.5.1 Grid world

We first tested our algorithm on the Grid-World MDP (Figure 2.2). It models an obstacle avoidance problem with stochastic dynamics. Each state corresponds to a square in the grid, and the actions, {N, NE, E, SE, S, SW, W, NW}, typically cause a move in the respective direction. In unmarked squares, the actor's intention is executed with probability .93. Each of the seven remaining actions might be executed instead, each with probability 0.01. Squares marked with `$` and `#` are absorbing states. The former gives a reward of 35 when entered, and the latter gives a penalty of 35. Any other state transitions cost 1. The horizon is 35. To make the problem finite, we simply set the probability of all transitions outside the grid boundary to zero, and re-normalize. We set the precision to $\varepsilon = 1$. With this setting, our algorithm performed exponential utility optimization for 97 different parameters when planning for 14 values of the risk level δ. For low values of δ, the algorithm behaves cautiously, preferring longer, but safer routes. For higher values of δ, the algorithm is willing to take shorter routes, but also accepts increasing amounts of risk.



(a) Cumulative distribution functions of rewards under Chernoff policies at different risk levels. The asterisk (*) indicates the value of the policy. The big O shows the expected reward and the small o's correspond to expectation plus-minus standard deviation. 10000 samples.

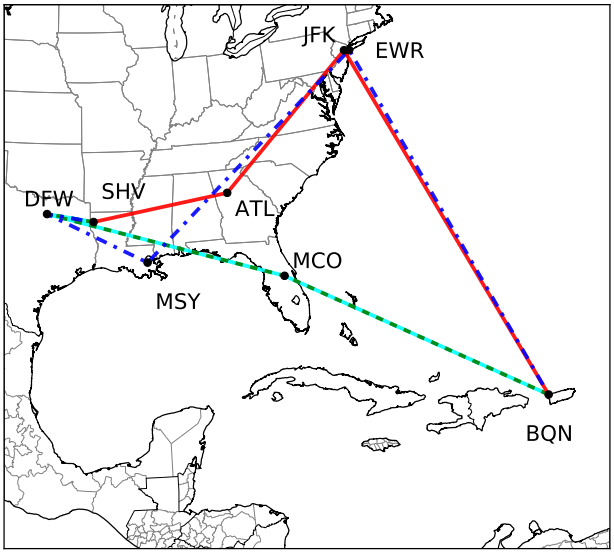
$\delta \in \{.99, .999, 1.0$ (expectimin)$\}$:
`15:45 SHV - DFW 16:45`
`18:25 DFW - MCO 21:50`
`23:15 MCO - BQN 02:46`

$\delta \in \{.3, .4, .5, .6, .7, .8, .9\}$:
`10:46 SHV - ATL 13:31`
`14:10 ATL - EWR 16:30`
`18:00 EWR - BQN 23:00`

$\delta = 0.2$:
`12:35 SHV - DFW 13:30`
`18:25 DFW - MCO 21:50`
`23:15 MCO - BQN 02:46`

$\delta \in \{0$ (minimax) $, .001, .01, .1\}$:
`12:35 SHV - DFW 13:30`
`14:25 DFW - MSY 15:50`
`17:50 MSY - JFK 21:46`
`23:40 JFK - BQN 04:20`

(b) Paths under Chernoff policies at different risk levels assuming all flight arrive on time.



(c) Map showing the different paths

Figure 2.3: Chernoff policies to travel from Shreveport Regional Airport (`SHV`) to Rafael Hernández Airport (`BQN`) at different risk levels. We are using International Air Transport Association (IATA) airport codes.

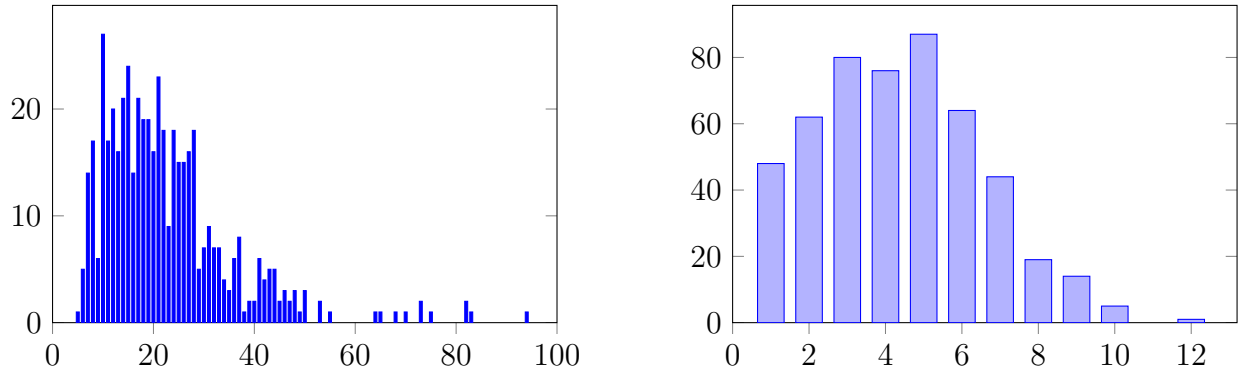
2.5.2 Air travel planning

The aerial travel planning MDP (Figure 2.3) illustrates that our method applies to real-world problems at a large scale. It models the problem of buying airplane tickets to travel between two cities, when you care only about reaching the destination in a reliable amount of time. We assume that, if you miss a connecting flight due to delays, the airline will re-issue a ticket for the route of your choice leading to the original destination. In this case, a cautious traveler will consider a number of aspects: choosing flights that usually arrive on time, choosing longer connection times and making sure that, in case of a missed connection, there are good alternative routes.

In our implementation, the state space consists of pairs of all airports and times when flights depart from those airports. At every state there are two actions: either take the flight that departs at that time, or wait. The total number of state-action pairs is 124791. To keep the horizon low, we introduce enough wait transitions so that it takes no more than 10 transitions to wait a whole day in the busiest airport (about 1000 flights per day) and we set the horizon at 100. Costs are deterministic and correspond to the time difference between the scheduled departure time of the first flight and the arrival time. We compute transition probabilities based on historical data, available from the Office of Airline Information, Bureau of Transportation Statistics, at `http://www.transtats.bts.gov/`. Particularly, we have used on-time statistics for February 2011. Airlines often try to conceal statistics for flights with low on-time performance by slightly changing departure times and flight numbers. Sometimes, they do this every week. Consequently, we first clustered together all flights with the same origin and destination that were scheduled to depart within 15 minutes of each other, under the assumption they would have the same on-time statistics. We, then, remove all clusters with less than 7 recorded flights, since these usually correspond to incidental flights.

To test our algorithm on this problem, we randomly chose 500 origin - destination airport pairs and computed the Chernoff policies for risk levels: $\delta \in \{1.0, .999, .99, .9, .8, .7, .6, .5, .4, .3, .2, .1, 0.01, 0.001, 0.0\}$, and precision $\varepsilon = 10$ minutes. Figure 2.3 shows the resulting policies and corresponding cost (travel time) histograms for one such randomly chosen route. To address the question of computational efficiency, Figure 2.4a shows a histogram of the total number of different parameters for which our algorithm ran exponential utility optimization. To address the question of relevance, Figure 2.4b shows the number of distinct Chernoff policies found among the risk levels. Two policies, π and π', are considered distinct if the total variation distance of the induced state - action occupation measures is more than 10^{-6}; that is, if there exists t, s, and a such that $|P_\pi\{S_t = s, A_t = a\} - P_{\pi'}\{S_t = s, A_t = a\}| \geq 10^{-6}$. For most origin - destination pairs we found a rich spectrum of distinct policies, but there are also cases where all the Chernoff policies are identical or only the expectimax and minimaxpolicies differ.

In general, we observed that lowering δ will produce policies with higher expected travel time, but lower standard deviation. However, in a few rare cases, we observed that lowering



(a) Number of exponential utility optimization runs to compute the Chernoff policies.

(b) Number of distinct Chernoff policies found.

Figure 2.4: Histograms demonstrating the efficiency and relevance of our algorithm on 500 randomly chosen origin - destination airport pairs, at 15 risk levels.

δ produces both higher expected travel time and higher standard deviations. This seems surprising if we only judge policies by mean and variance, however looking at histograms showed that the lower δ policies avoided extremely bad outcomes, at the cost of higher variance. In this case, our algorithm automatically traded performance in the mean minus variance sense for performance in the percentile sense. Intuitively, such trade-offs are necessary whenever the set of allowed policies is small, which is the case in this MDP.

2.6 Discussion

We proposed a new optimization objective for risk-aware planning called the Chernoff functional. Our objective has a free parameter δ that can be used to interpolate between the nominal policy, which ignores risk, at $\delta = 1$, and the minmax policy, which corresponds to extreme risk aversion, at $\delta = 0$. The value of the Chernoff functional is with probability at least $1 - \delta$ an upper bound on the cost incurred by following the corresponding Chernoff policy. We established a close connection between optimizing the Chernoff functional and mean minus variance optimization, which has been proposed before for risk-aware planning, but was found to be intractable in general. We also establish a close connection with optimization of mean minus scaled standard deviation.

We proposed an efficient algorithm that optimizes the Chernoff functional to any desired accuracy ε requiring $O(1/\varepsilon)$ runs of exponential utility optimization. Our experiments illustrate the capability of our approach to recover a spread of policies in the spectrum from risk neutral to minmax requiring a running time that was on average about ten times the running of value iteration.

Chapter 3

Safe Exploration for Markov Decision Processes

3.1 Introduction

When humans learn to control a system, they naturally account for what we think of as safety. For example, when a novice pilot learns how to fly an RC helicopter, they will slowly spin up the blades until the helicopter barely lifts off, then quickly put it back down. They will repeat this a few times, slowly starting to bring the helicopter a little bit off the ground. When doing so they would try out the cyclic (roll and pitch) and rudder (yaw) control, while—until they have become more skilled—at all times staying low enough that simply shutting it down would still have it land safely. When a driver wants to become skilled at driving on snow, they might first slowly drive the car to a wide open space where they could start pushing their limits. When we are skiing downhill, we are careful about not going down a slope into a valley where there is no lift to take us back up.

One would hope that exploration algorithms for physical systems would be able to account for safety and have similar behavior naturally emerge. Unfortunately most existing exploration algorithms completely ignore safety issues. More precisely phrased, most existing algorithms have strong exploration guarantees, but to achieve these guarantees they assume ergodicity of the Markov decision process (MDP) in which the exploration takes place. An MDP is *ergodic* if any state is reachable from any other state by following a suitable policy. This assumption does not hold true in the exploration examples presented above as each of these systems could break during (non-safe) exploration.

The first contribution of this Chapter is a definition of safety, which, at its core, requires restricting attention to policies that preserve ergodicity with some well controlled probability. Imposing safety is, unfortunately, NP-hard in general. The second contribution of the Chapter is an approximation scheme leading to guaranteed safe, but potentially sub-optimal, exploration. Note that existing (unsafe) exploration algorithms are also sub-optimal, in that

they are not guaranteed to complete exploration in the minimal number of time steps. A third contribution is the consideration of uncertainty in the dynamics model that is correlated over states. While usually the assumption is that uncertainty in different parameters is independent—as this makes problem more tractable computationally—being able to learn about state-action pairs before visiting them is critical for safety.

Our experiments illustrate that our method indeed achieves safe exploration, in contrast to exploration methods that depend on the ergodicity assumption. They also show that our algorithm is almost as computationally efficient as planning in a known MDP—but then, as every step leads to an update in knowledge about the MDP, this computation is to be repeated after every step. Our approach is able to safely explore grid worlds of size up to 50×100. Our method can make safe any type of exploration that relies on exploration bonuses, which is the case for most existing exploration algorithms, including, for example, the methods proposed in [22, 43]. In this Chapter we do not focus on the exploration objective and use existing ones.

Safe exploration has been the focus of a large number of articles. [36, 9] propose safe exploration methods for linear systems with bounded disturbances based on model predictive control and reachability analysis. They define safety in terms of safe regions of the state space, which, we will show, is not always appropriate in the context of MDPs. The safe exploration for MDP methods proposed by [35, 37] gauge safety based on the best estimate of the transition measure but they ignore the level of uncertainty in this estimate. As we will show, this is not sufficient to provably guarantee safety.

Provably efficient exploration is a recurring theme in reinforcement learning [60, 47, 22, 41, 43]. Most methods, however, tend to rely on the assumption of ergodicity which rarely holds in interesting practical examples; consequently, these methods are rarely applicable for physical systems. The issue of provably guaranteed safety, or risk aversion, under uncertainty in the MDP parameters has also been studied in the reinforcement literature. In [55] they propose a robust MDP control method assuming the transition frequencies are drawn from an orthogonal convex set by an adversary. Unfortunately, it seems impossible to use their method to constrain some safety objective while optimizing a different exploration objective. In [31] they present a safe exploration algorithm for the special case of Gaussian distributed ambiguity in the reward and state-action-state transition probabilities, but their safety guarantees are only accurate if the ambiguity in the transition model is small.

This Chapter follows the previously published article by Moldovan and Abbeel [53] with minor extensions.

3.2 Notation and Assumptions

For an introduction to Markov Decision Processes (MDPs) we refer the readers to [62, 15]. In this chapter, we use capital letters to denote random variables; for example, the total reward is: $V := \sum_{t=0}^{\infty} R_{S_t,A_t}$. We represent the policies and the initial state distributions by

probability measures. Usually the measure π will correspond to a policy and the measure $s := \delta(s)$, which puts measure only in state s, will correspond to starting in state s. With this notation, the usual value recursion, assuming a known transition measure, p, reads:

$$E^p_{s,\pi}[V] = \sum_{a,s'} \pi_{s,a} \left(E[R]_{s,a} + p_{s,a,s'} E^p_{s',\pi}[V] \right).$$

We specify the transition measure as a superscript of the expectation operator rather than a subscript for typographical convenience; in this case, and in general, the positioning of indexes as subscripts or superscripts adds no extra significance. We will let the transition measure p sometimes sum to less than one, that is $\sum_{s'} p_{s,a,s'} \leq 1$. The missing mass is implicitly assigned to transitioning to an absorbing "end" state, which, for example, allows us to model γ discounting by simply using γp as a transition measure.

We model ambiguous dynamics in a Bayesian way, allowing the transition measure to also be a random variable. When this is the case, we will use P to denote the, now random, transition measure. The *belief*, which we will denote by β, is our Bayesian probability measure over possible dynamics, governing P and R. Therefore, the expected return under the belief and policy π, starting from state s, is $E_\beta E^P_{s,\pi}[V]$. We allow beliefs under which transition measures and rewards are arbitrarily correlated. In fact, such correlations are usually necessary to allow for safe exploration. For compactness we will often use lower case letters to denote the expectation of their upper case counterparts. Specifically, we will use the notations $p := E_\beta[P]$ and $r := E_\beta[R]$ throughout.

3.3 Problem formulation

3.3.1 Exploration Objective

Exploration methods, as those proposed in [22, 43], operate by finding optimal policies in constructed MDPs with exploration bonuses. The R-MAX algorithm, for example, constructs an MDP based on the discounted expected transition measure and rewards under the belief, and adds a deterministic exploration bonus equal to the maximum possible reward in the MDP, $\xi^\beta_{s,a} = r_{\max}$, to any transitions that are not sufficiently well known. Our method allows adding safety constraints to any such exploration methods. Henceforth, we will restrict attention to such exploration methods, which can be formalized as optimization problems of the form:

$$\text{maximize}_{\ \pi_o}\ E^{\gamma p}_{s_0,\pi_o} \sum_{t=0}^{\infty} \left(r_{S_t,A_t} + \xi^\beta_{S_t,A_t} \right). \tag{3.1}$$

3.3.2 Safety Constraint

The issue of safety is closely related to *ergodicity*. Almost all proposed exploration techniques presume ergodicity; authors present it as a harmless technical assumption but it rarely holds in interesting practical problems. Whenever this happens, their efficient exploration guarantees cease to hold, often leading to very inefficient policies. Informally, an environment is ergodic if any mistake can be forgiven eventually. More specifically, a belief over MDPs is ergodic if and only if any state is reachable from any other state via some policy or, equivalently, if and only if:

$$\forall s, s', \exists\ \pi_r \text{ such that } E_\beta E^P_{s,\pi_r}[B_{s'}] = 1, \tag{3.2}$$

where $B_{s'}$ is an indicator random variable of the event that the system reaches state s' at least once: $B_{s'} = 1\{\exists t < \infty \text{ such that } S_t = s'\} = \min\left(1, \sum_t 1_{S_t = s'}\right)$.

Unfortunately, many environments are not ergodic. For example, our robot helicopter learning to fly cannot recover on its own after crashing. Ensuring almost sure ergodicity is too restrictive for most environments as, typically, there always is a very small, but non-zero, chance of encountering that particularly unlucky sequence of events that breaks the system. Our idea is to restrict the space of eligible policies to those that preserve ergodicity with some user-specified probability, δ, called the *safety level*. We name these policies δ*-safe*. Safe exploration now amounts to choosing the best exploration policy from this set of safe policies.

Informally, if we stopped a δ-safe policy π_o at any time T, we would be able to return from that point to the home state s_0 with probability δ by deploying a return policy π_r. Executing only δ-safe policies in the case of a robot helicopter learning to fly will guarantee that the helicopter is able to land safely with probability δ whenever we decide to end the experiment. In this example, T is the time when the helicopter is recalled (perhaps because fuel is running low), so we will call T the *recall time*. Formally, an outbound policy π_o is δ-safe with respect to a home state s_0 and a stopping time T if and only if:

$$\exists \pi_r \text{ such that } E_\beta E^P_{s_0,\pi_o}\left[E^P_{S_T,\pi_r}[B_{s_0}]\right] \geq \delta. \tag{3.3}$$

Note that, based on Equation (3.2), any policy is δ-safe for any δ if the MDP is ergodic with probability one under the belief. For convenience we will assume that the recall time, T, is exponentially distributed with parameter $1-\gamma$, but our method also applies when the recall time equals some deterministic horizon. Unfortunately, expressing the set of δ-safe policies is NP-hard in general, as implied by the following theorem proven in the appendix.

Theorem 3. *In general, it is NP-hard to decide whether there exist δ-safe policies with respect to a home state, s_0, and a stopping time, T, for some belief, β.*

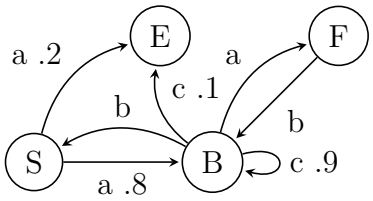


Figure 3.1: Starting from state S, the policy (aababab...) is safe at a safety level of .8. However, the policy (acccc...) is not safe since it will end up in the sink state E with probability 1. State-action Sa and state B can neither be considered safe nor unsafe, since both policies use them.

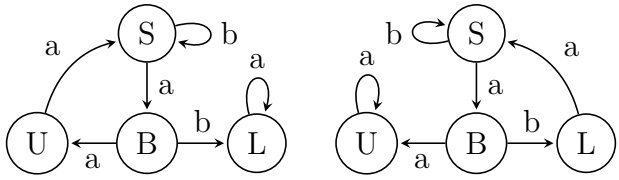


Figure 3.2: Under out belief the two MDPs above both have probability .5. It is intuitively unsafe to go from the start state S to B since we wouldn't know whether the way back is via U or L, even though we know for sure that a return policy exists.

3.3.3 Safety Counter-Examples

We conclude this section with counter-examples to three other, perhaps at first sight more intuitive, definitions of safety. First, we could have tried to define safety in terms of sets of safe states or state-actions. That is, we might think that making the non-safe states and actions unavailable to the planner (or simply inaccessible) is enough to guarantee safety. Figure 3.1 shows an MDP where the same state-action is used both by a safe and by an unsafe policy. The idea behind this counter-example is that safety depends not only on the states visited, but also on the number of visits, thus, on the policy. This shows that safety should be defined in terms of safe policies, not in terms of safe states or state-actions.

Second, we might think that it is perhaps enough to ensure that there exists a return policy for each potential sample MDP from the belief, but not impose that it be the same for all samples. That is, we might think that condition (3.3) is too strong and, instead, it would be enough to have:

$$E_\beta 1\{\exists \pi_r : E^P_{s_0,\pi_o} E^P_{S_T,\pi_r}[B_{s_0}] = 1\} \geq \delta.$$

Figure 3.2 shows an MDP where this condition holds, yet all policies are naturally unsafe.

Third, we might think that it is sufficient to simply use the expected transition measure when defining safety, as in the equation below. Figure 3.3 shows that this is not the case;

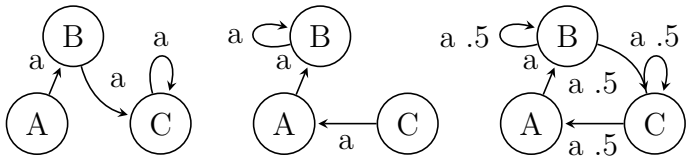


Figure 3.3: The two MDPs on the left both have probability .5. Under this belief, starting from state A, policy (aaa...) is unsafe. However, under the mean transition measure, represented by the MDP on the right, the policy is safe.

the expected transition measure is not a sufficient statistic for safety.

$$\exists \pi_r \text{ such that } E^p_{s_0,\pi_o}\left[E^p_{S_T,\pi_r}[B_{s_0}]\right] \geq \delta.$$

3.4 Guaranteed Safe, Potentially Sub-optimal Exploration

Although imposing the safety constraint in Equation (3.3) is NP-hard, as shown in Theorem 3, we can efficiently constrain a lower bound on the safety objective, so the safety condition is still provably satisfied. Doing so could lead to sub-optimal exploration since the set of policies we are optimizing over has shrunk. However, we should keep in mind that the exploration objectives represent approximate solutions to other NP-hard problems, so optimality has already been forfeited in existing (non-safe) approaches to start out with. Algorithm 2 summarizes the procedure and the experiments presented in the next section show that, in practice, when the ergodicity assumptions are violated, safe exploration is much more efficient than plain exploration.

Putting together the exploration objective defined in Equation (3.1) and the safety objective defined in Equation (3.3) allows us to formulate safe exploration at level δ as a constrained optimization problem:

$$\begin{aligned} \text{maximize } {}_{\pi_o,\pi_r} \quad & E^{\gamma p}_{s_0,\pi_o} \sum_t \left(r_{S_t,A_t} + \xi^{\beta}_{S_t,A_t} \right) \\ \text{such that:} \quad & E_\beta E^P_{s_0,\pi_o}\left[E^P_{S_T,\pi_r}[B_{s_0}]\right] \geq \delta. \end{aligned}$$

The exploration objective is already conveniently formulated as the expected reward in an MDP with transition measure γp, so we will not modify it. On the other hand, the safety constraint is difficult to deal with as is. Ideally, we would like the safety constraint to also equal some expected reward in an MDP. We will see that, in fact, it takes two MDPs to express the safety constraint.

First, we express the inner term, $E^P_{S_T,\pi_r}[B_{s_0}]$, as the expected reward in an MDP. We can replicate the behaviour of B_{s_0}, that is counting only the first time state s_0 is reached,

Algorithm 2 Safe exploration algorithm

Require: prior belief β, discount γ, safety level δ.
Require: function ξ : belief $\to$ exploration bonus
  $M, N \leftarrow$ new MDP objects
  repeat
    $s_0, \varphi \leftarrow$ current state and observations
    update belief β with information φ
    $\xi^{\beta}_{s,a} \leftarrow \xi(\beta)$ (exploration bonus based on β)
    $\sigma^{\beta}_{s,a} \leftarrow \sum_{s'} E_{\beta}[\min(0, P_{s,a,s'} - E_{\beta}[P_{s,a,s'}])]$
    M.transition measure $\leftarrow E_{\beta}[P](1 - 1_{s=s_0})$
    M.reward function $\leftarrow 1_{s=s_0} + (1 - 1_{s=s_0})\sigma^{\beta}_{s,a}$
    $\pi^r, v \leftarrow M$.solve()
    N.transition measure $\leftarrow \gamma E_{\beta}[P]$
    N.reward function $\leftarrow E_{\beta}[R_{s,a}] + \xi^{\beta}_{s,a}$
    N.constraint reward func. $\leftarrow (1-\gamma)v_s + \gamma\sigma^{\beta}_{s,a}$
    N.constraint lower bound $\leftarrow \delta$
    $\pi^o, v^{\xi}, v^{\sigma} \leftarrow N$.solve under constraint()
    $q^{\sigma}_{s,a} \leftarrow (1-\gamma)v_s + \gamma\sigma^{\beta}_{s,a} + \sum_{s'} p_{s,a,s'} v^{\sigma}_{s'}$
    $a \leftarrow \text{argmax}_{\{\pi^o_{s_0,a} > 0\}} q^{\sigma}_{s_0,a}$ (de-randomize policy)
    take action a in environment
  until $\xi^{\beta} = 0$, so there is nothing left to explore

by using a new transition measure, $P \cdot (1 - 1_{s=s_0})$ under which, once s_0 is reached, any further actions lead immediately to the implicit "end" state. Formally, we express this by the identity:

$$E^{P}_{S_T,\pi_r}[B_{s_0}] = E^{P\cdot(1-1_{s=s_0})}_{S_T,\pi_r} \sum_{t=0}^{\infty} 1_{S_t=s_0}.$$

We now focus on the outer term, $E^{P}_{s_0,\pi_o}\left[E^{P}_{S_T,\pi_r}[B_{s_0}]\right]$. Since the recall time, T, is exponentially distributed with parameter $1-\gamma$, we can view S_T as the final state in a γ-discounted MDP starting at state s_0, following policy π_o. In this MDP, the inner term plays the role of a terminal reward. To put the problem in a standard form, we convert this terminal reward to a step-wise reward by multiplying it by $1-\gamma$.

$$E_{\beta}E^{P}_{s_0,\pi_o}\left[E^{P}_{S_T,\pi_r}[B_{s_0}]\right] = E_{\beta}E^{\gamma P}_{s_0,\pi_o} \sum_{t=0}^{\infty}(1-\gamma)\left[E^{P}_{S_t,\pi_r}[B_{s_0}]\right].$$

At this point, we have expressed the safety constraint in the MDP formalism, but the transition measures of these MDPs, $P(1 - 1_{s=s_0})$ and γP, are still random. If we could

replace these random transition measures with their expectations under the belief β that would significantly simplify the safety constraint. It turns out we can do this, at the expense of making the constraint more stringent. Our tool for doing so is Theorem 4 presented below, but proven in the appendix. It shows that we can replace a belief over MDPs by a single MDP with the expected transition measure, featuring an appropriate reward correction such that, under any policy, the value of this MDP is a lower bound on the expected value under the belief.

Theorem 4. *Let β be a belief such that for any policy, π, and any starting state, s, the total expected reward in any MDP drawn from the belief is between 0 and 1; i.e. $0 \leq E^P_{s,\pi}[V] \leq 1$, β-almost surely. Then the following bound holds for any policy, π, and any starting state, s:*

$$E_\beta E^P_{s,\pi} \sum_{t=0}^{\infty} R_{S_t,A_t} \geq E^p_{s,\pi} \sum_{t=0}^{\infty} \left(E_\beta[R_{S_t,A_t}] + \sigma^\beta_{S_t,A_t} \right)$$
$$\textit{where} \quad \sigma^\beta_{s,a} := \sum_{s'} E_\beta \left[\min(0, P_{s,a,s'} - E_\beta[P_{s,a,s'}]) \right].$$

We first apply Theorem 4 to the outer term, yielding the following bound:

$$E_\beta E^P_{s_0,\pi_o} \left[E^P_{S_T,\pi_r}[B_{s_0}] \right] = E_\beta E^{\gamma P}_{s_0,\pi_o} \sum_{t=0}^{\infty} (1-\gamma) E^P_{S_t,\pi_r}[B_{s_0}]$$
$$\geq E^{\gamma p}_{s_0,\pi_o} \sum_{t=0}^{\infty} \left((1-\gamma) E_\beta E^P_{S_t,\pi_r}[B_{s_0}] + \gamma \sigma^\beta_{S_t,A_t} \right).$$

We, then, apply it again to the inner term:

$$E_\beta E^P_{s,\pi_r}[B_{s_0}] = E^{P \cdot (1 - 1_{s=s_0})}_{s,\pi_r} \sum_{t=0}^{\infty} 1_{S_t = s_0} \geq \tag{3.4}$$
$$\geq E^{p \cdot (1-1_{s=s_0})}_{s,\pi_r} \sum_{t=0}^{\infty} \left(1_{S_t=s_0} + (1 - 1_{S_t=s_0}) \sigma^\beta_{S_t,A_t} \right).$$

Combining the last two results allows us to replace the NP-hard safety constraint with a stricter, but now tractable, constraint. The resulting optimization problem corresponds to the guaranteed safe, but potentially sub-optimal exploration problem:

$$\text{maximize}_{\ \pi_o,\pi_r} \quad E^{\gamma p}_{s_0,\pi_o} \sum_t \left(r_{S_t,A_t} + \xi^\beta_{S_t,A_t} \right) \tag{3.5}$$
$$\text{s.t.:} \quad E^{\gamma p}_{s_0,\pi_o} \sum_{t=0}^{\infty} \left((1-\gamma) v_{S_t} + \gamma \sigma^\beta_{S_t,A_t} \right) \geq \delta \quad \text{and}$$
$$v_s = E^{p \cdot (1-1_{s=s_0})}_{s,\pi_r} \sum_{t=0}^{\infty} \left(1_{S_t=s_0} + (1 - 1_{S_t=s_0}) \sigma^\beta_{S_t,A_t} \right).$$

The term v_s represents our lower bound for the inner term per Equation (3.4), and is simply the value function of the MDP corresponding to the inner term; i.e. the MDP with transition measure $p(1 - 1_{s=s_0})$ and reward function $1_{s=s_0} + (1 - 1_{s=s_0})\sigma^{\beta}_{s,a}$, under policy π_r. Since the return policy, π_r, does not appear anywhere else, we can split the safe exploration problem we obtained in Equation (3.5) into two steps:

Step one: find the optimal return policy π^*_r, and corresponding value function v^*_s, by solving the standard MDP problem below:

$$E^{p\cdot(1-1_{s=s_0})}_{s,\pi_r} \sum_{t=0}^{\infty} \left(1_{S_t=s_0} + (1 - 1_{S_t=s_0})\sigma^{\beta}_{S_t,A_t}\right). \tag{3.6}$$

Step two: find the optimal exploration policy π^*_o under the strict safety constraint, by solving the constrained MDP problem below:

$$\text{maximize}_{\ \pi_o} \quad E^{\gamma p}_{s_0,\pi_o} \sum_t \left(r_{S_t,A_t} + \xi^{\beta}_{S_t,A_t}\right) \tag{3.7}$$

$$\text{s.t.:} \quad E^{\gamma p}_{s_0,\pi_o} \sum_{t=0}^{\infty} \left((1-\gamma)v^*_{S_t} + \gamma\sigma^{\beta}_{S_t,A_t}\right) \geq \delta. \tag{3.8}$$

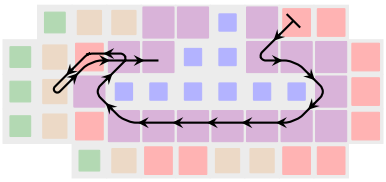
The first step amounts to solving a standard MDP problem while the second step amounts to solving a constrained MDP problem. As shown by [6], both can be solved efficiently either by linear programming, or by value-iteration. In our experiments we used the LP formulation with the state-action occupation measure as optimization variable. Solutions to the constrained MDP problem will usually be stochastic policies, and, in our experiments, we found that following them sometimes leads to random walks which explore inefficiently. We addressed the issue by de-randomizing the exploration policies in favor of safety. That is, whenever the stochastic policy proposes multiple actions with non-zero measure, we choose the one among them that optimizes the safety objective.

3.5 Experiments

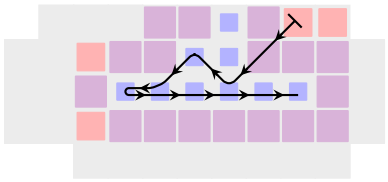
3.5.1 Grid World

Our first experiment models a terrain exploration problem where the agent has limited sensing capabilities. We consider a simple rectangular grid world, where every state has a height H_s. From our Bayesian standpoint these heights are independent, uniformly distributed categorical random variables on the set $\{1, 2, 3, 4, 5\}$. At any time the agent can attempt to move to any immediately neighboring state. Such move will succeed with probability one if the height of the destination state is no more than one level above the current state;

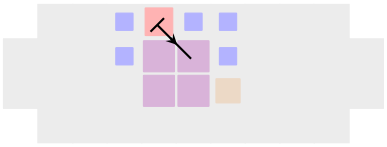
(a) Based on the available information after the first step, moving South-West is unsafe.



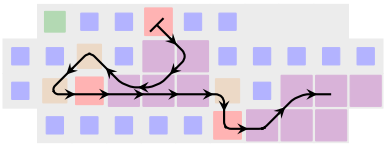
(b) The safe explorer successfully uncovers all of the map by avoiding irreversible actions.



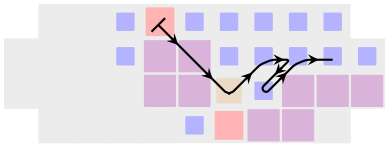
(c) The adapted R-MAX explorer gets stuck before observing the entire map.



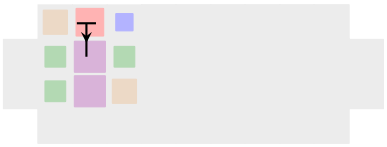
(d) Moving South-East is currently considered unsafe since, based on the available information, there is no return path.



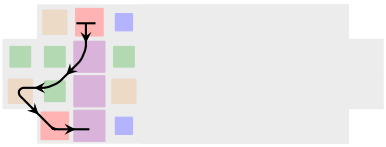
(e) After seeing more of the map, our safe explorer decides that the transition initially deemed unsafe is, in fact, safe.



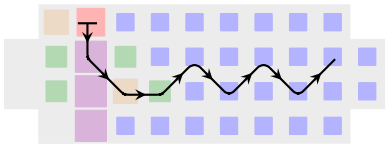
(f) The adapted R-MAX explorer acts greedily. Even though its second action, is, in fact safe, its third action is not, so it gets stuck.



(g) Moving East is safe with probability .8 since the return path is blocked for only one out of five possible heights of the unknown square South of the start position.



(h) Safe exploration with $\delta = 1.0$ does not risk moving East event though the exploration bonuses are much higher there.



(i) Safe exploration with $\delta \leq .6$ does move East. Note that, in this case, our method overestimates the probability of failure by a factor of two and, thus, acts conservatively.

Figure 3.4: Exploration experiments in simple grid worlds. See text for full details. Square sizes are proportional to corresponding state heights between 1 and 5. The large, violet squares have a height of 5, while the small, blue squares have a height of 1. Gray spaces represent states that have not yet been observed. Each row corresponds to the same grid world. The first column shows the belief after the first exploration step, while the second and third columns show the entire trajectory followed by different explorers.

otherwise, the agent remains in the current state with probability one. In other words, the agent can always go down cliffs, but is unable to climb up if they are too steep. Whenever the agent enters a new state it can see the exact heights of all immediately surrounding states. We present this grid world experiment to build intuition and to provide an easily reproducible result. Figure 3.4 shows a number of examples where our exploration method results in intuitively safe behavior, while plain exploration methods lead to clearly unsafe, suboptimal behavior.

Our exploration scheme, which we call *adapted* R-MAX, is a modified version of R-max exploration [22], where the exploration bonus of moving between two states is now proportional to the number of neighboring unknown states that would be uncovered as a result of the move, to account for the remote observation model. The safety costs for this exploration setup, as prescribed by Theorem 4 are:

$$\sigma^{\beta}_{s,a} = -2E_{\beta}[P_{s,a}](1 - E_{\beta}[P_{s,a}]) = -2\mathrm{Var}_{\beta}[P_{s,a}]$$

where $P_{s,a} := 1_{H_{s+a} \leq H_s + 1}$ is the probability that attempted move a succeeds in state s and the belief β describes the distribution of the heights of unseen states. In practice we found that this correction is a factor of two larger than would be sufficient to give a tight safety bound.

A somewhat counter intuitive result is that adding safety constraints to the exploration objective will, in fact, improve the fraction of squares explored in randomly generated grid worlds. The reason why plain exploration performs so poorly is that the ergodicity assumptions are violated, so efficiency guarantees no longer hold. Figure 3.5 summarizes our exploration performance results.

3.5.2 Martian Terrain

For our second experiment, we model the problem of autonomously exploring the surface of Mars by a rover such as the *Mars Science Laboratory (MSL)* [49]. The MSL is designed to be remote controlled from Earth but communication suffers a latency of 16.6 minutes. At top speed, it could traverse about 20m before receiving new instructions, so it needs to be able to navigate autonomously. In the future, when such rovers become faster and cheaper to deploy, the ability to plan their paths autonomously will become even more important. The MSL is designed to a static stability of 45 degrees, but would only be able to climb slopes up to 5 degrees without slipping [1]. Digital terrain models for parts of the surface of Mars are available from the *High Resolution Imaging Science Experiment (HiRISE)* at a scale of 1.00 meter/pixel and accurate to about a quarter of a meter. The MSL would be able to obtain much more accurate terrain models locally by stereo vision.

The state-action space of our model MDP is the same as in the previous experiment, with each state corresponding to a square area of 20 by 20 meters on the surface. We allow only transitions at slopes between -45 and 5 degrees. The heights, H_s, are now assumed to

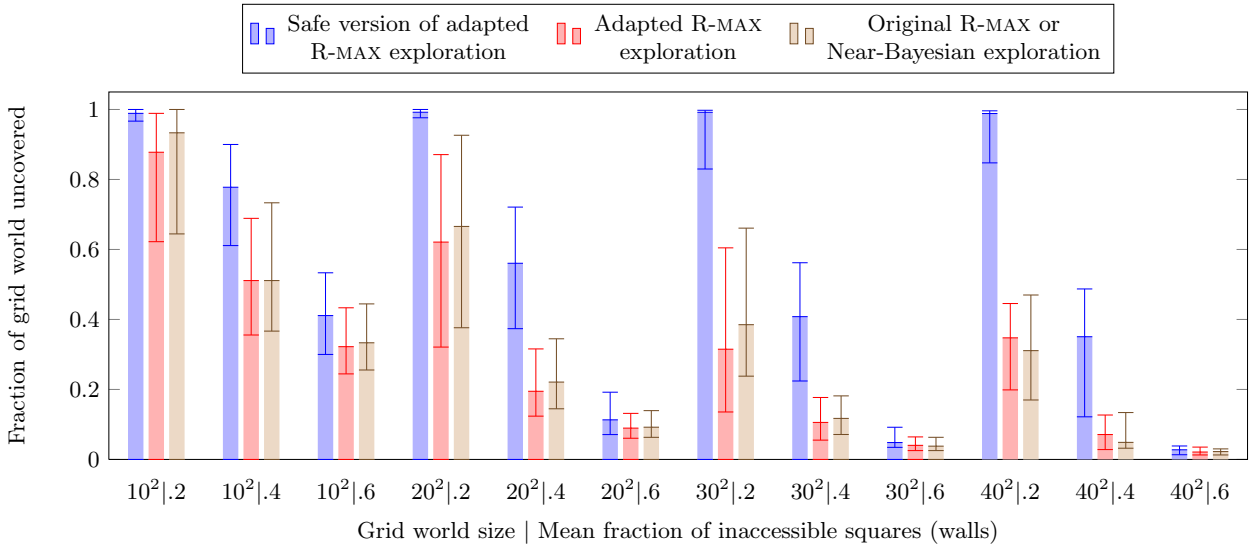
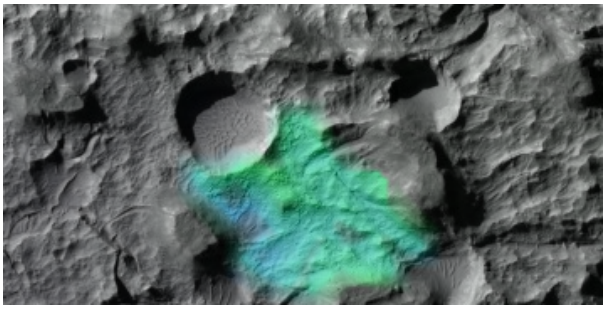
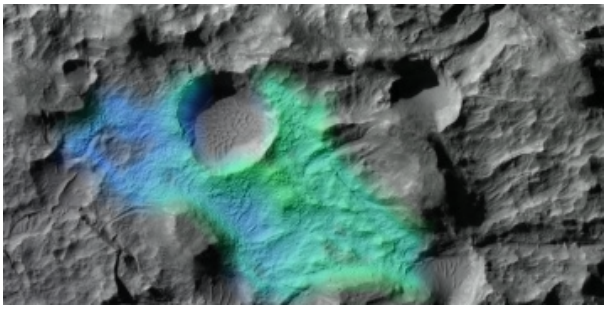


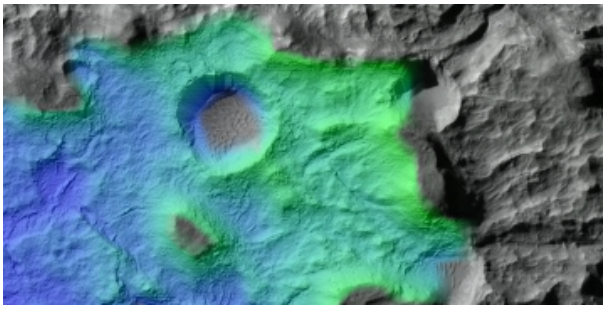
Figure 3.5: Exploration efficiency comparison. We are showing the median, the upper and the lower quartiles of the fraction of the grid world that was uncovered by different explorers in randomly generated grid worlds. The "amount" of non-ergodicity is controlled by randomly making a fraction of the squares inaccessible (walls). We ran 1000, 500, 100,20 experiments for grids of sizes 10^2, 20^2, 30^2 and 40^2 respectively. We are comparing against our own adapted R-MAX exploration objective, the original R-MAX objective [22] and the Near-Bayesian exploration objective [43]. The last two behave identically in our grid world environment, since, once a state is visited, all transitions out of that state are precisely revealed.



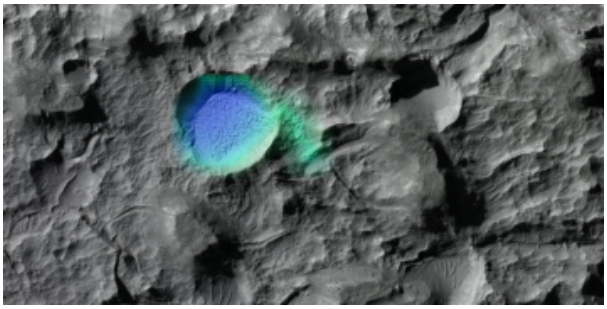
(a) Safe exploration with $\delta = .98$ leads to a model entropy reduction of 7680.



(b) Safe exploration with $\delta = .90$ leads to a model entropy reduction of 12660.



(c) Safe exploration with $\delta = .70$ leads to a model entropy reduction of 35975.



(d) Regular (unsafe, $\delta = 0$) exploration leads to a model entropy reduction of 3214.

Figure 3.6: Simulated safe exploration on a 2km by 1km area of Mars at -30.6 degree latitude and 202.2 degrees longitude, for 15000 time steps, at different safety levels. See text for full details. The color saturation is inversely proportional to the standard deviation of the height map under the posterior belief. Full coloration represents a standard deviation of 1cm or less. We report the difference between the entropies of the height model under the prior and the posterior beliefs as a measure of performance. Images: NASA/JPL/University of Arizona.

Table 3.1: Per-step planning times for the 50×100 grid world used in the Mars exploration experiments, with $\gamma = .999$.

Problem setting	Planning time (s)
Safe exploration at .98	5.86 ± 1.47
Safe exploration at .90	10.94 ± 7.14
Safe exploration at .70	4.57 ± 3.19
Naive constraint at .98	2.55 ± 0.42
Regular (unsafe) exploration	1.62 ± 0.26

be independent Gaussian random variables. Under the prior belief, informed by the HiRISE data, the expected heights and their variances are:

$$E_\beta[H] = D^{20}[g \circ h] \quad \text{and} \quad \mathrm{Var}_\beta[H] = D^{20}[g \circ (h - g \circ h)^2] + v_0$$

where h are the HiRISE measurements, g is a Gaussian filter with $\sigma = 5$ meters, "$\circ$" represents image convolution, D^{20} is the sub-sampling operator and $v_0 = 2^{-4}\mathrm{m}^2$ is our estimate of the variance of HiRISE measurements. We model remote sensing by assuming that the MSL can obtain Gaussian noisy measurements of the height at a distance d away with variance $v(d) = 10^{-6}(d + 1\mathrm{m})^2$. To account for this remote sensing model we use a first order approximation of the entropy of H as an exploration bonus:

$$\xi^\beta_{s,a} = \sum_{s'} \mathrm{Var}_\beta[H_{s'}]/v(d_{s,s'}).$$

Figure 3.6 shows our simulated exploration results for a 2km by 1km area at -30.6 degrees latitude and 202.2 degrees longitude [2]. Safe exploration at level 1.0 is no longer possible, but, even at a conservative safety level of .98, our method covers more ground than the regular (unsafe) exploration method which promptly get stuck in a crater. Imposing the safety constraint naively, with respect to the expected transition measure, as argued against at the end of Section 3.3.3, performs as poorly as unsafe exploration even if the constraint is set at .98.

3.5.3 Computation Time

We implemented our algorithm in Python 2.7.2.7, using Numpy 1.5.1 for dense array manipulation, SciPy 0.9.0 for sparse matrix manipulation and Mosek 6.0.0.119 for linear programming. The discount factor was set to .99 for the grid world experiment and to .999 for Mars exploration. In the latter experiment we also restricted precision to 10^{-6} to avoid numerical instabilities in the LP solver. Table 3.1 summarizes planning times for our Mars exploration experiments.

3.6 Discussion

We provided a formalization of the problem of safe exploration and showed that, in general, imposing safety in the least constrictive way is NP-hard. In response to this limitation, we proposed an efficient algorithm that is guaranteed to be safe, but potentially sub-optimal. Its running time is on par with existing exploration algorithms, such as R-MAXand near-Bayesian exploration, which ignore safety and which are also sub-optimal. Our method relies on the assumption that knowledge about already visited state-actions is informative of to neighboring state-action pairs; this assumption, we argued, is critical for safety. Experiments illustrate that our approach indeed achieves safe exploration, in contrast to exploration methods designed for ergodic environments only. Our approach decouples the exploration objective from the safety constraints. Consequently, our method can make any type of exploration safe if the incentive to explore is encoded in the reward, as is the case for most existing exploration algorithms.

In addition to the safety formulation we discussed in Section 3.3.2, out framework also supports a number of other safety criteria summarized below. Any combination of these constraints can be imposed simultaneously by appending the corresponding strict approximation to the objective expressed in Equation (3.7). A stricter ergodicity constraint would ensure that return is possible within some horizon, h, not just eventually, with probability δ. In this case Theorem 4 needs to be extended to the finite horizon case and the MDP defined by Equation (3.6) needs to be replaced as follows:

$$E_{s,\pi_r}^{p\cdot(1-1_{s=s_0})} \sum_{t=0}^{h} \left(1_{S_t=s_0} + (1-1_{S_t=s_0})\sigma^{\beta}_{S_t,A_t}\right).$$

Alternatively, we could ensure that the probability of leaving some pre-defined safe set of state-actions $\mathcal{G}$ is lower than $1-\delta$. In this case, the step specified by Equation (3.6) is no longer required and the constrained specified by Equation (3.8) needs to be replaced as follows:

$$E_{s_0,\pi_o}^{\gamma p} \sum_{t=0}^{\infty} \left((1-\gamma)1_{(S_t,A_t)\in\mathcal{G}} + \gamma\sigma^{\beta}_{S_t,A_t}\right) \geq \delta.$$

Finally we could ensure that the expected total reward under the belief is higher than δ assuming that the reward r is bounded by $0 \leq r \leq 1-\gamma$. In this case, the step specified by Equation (3.6) is no longer required and the constrained specified by Equation (3.8) needs to be replaced as follows:

$$E_{s_0,\pi_o}^{\gamma p} \sum_{t=0}^{\infty} \left(r_{S_t,A_t} + \gamma\sigma^{\beta}_{S_t,A_t}\right) \geq \delta.$$

Chapter 4

Exploration for Continuous Dynamical Systems

4.1 Introduction

Model-based reinforcement learning considers the general problem of learning to control systems with dynamics and reward functions not known in advance, that need to be learned by exploratory control. In this Chapter we focus on the problem of unknown dynamics, and assume the reward function is known. The key challenges in in this setting are: (i) Modeling the unknown dynamics from data, (ii) Performing effective exploration, ideally taking advantage of the fact that the goal state is known ahead of time, and hence exploration can be goal-directed rather than needing to explore the entire space, and (iii) Efficiently planning a sequence of actions for the current dynamics model (which includes uncertainty) and exploration strategy. To address these challenges we bring to bear recent advances in non-parametric supervised learning for modeling, advances in model-based reinforcement learning for exploration, and advances in nonlinear optimization and pseudo-spectral methods for optimal control. Our contributions consist of adapting these components as necessary and combining them into a system that shows state-of-the-art learning performance empirically. Each component is discussed separately below.

Modeling: From a modeling perspective, the problem is difficult because no single parametric model will apply to all situations; it would either over-fit in some cases or not be rich enough in other cases. Ideally, the complexity of the model should increase as more information about the system dynamics becomes available. Gaussian process dynamic programming [30, 28] works by modeling both the dynamics and the value function as Gaussian processes, but, in practice, the cubic time complexity of this approach only allows a small fraction of the training data to be used. On-line versions of Gaussian Processes have also been proposed but they require further approximations. As an alternative we propose using the Dirichlet process mixture of Gaussians. Our first contribution of this Chapter is em-

pirically validating the applicability of this model. The mixture model can be trained in (expected) time $O(n \log n)$ where n is the number of observations allowing us to use larger amounts of training data. This model has been used in robotics application, but only in the context of learning from demonstrations [24]. Section 4.2 describes the Bayesian Linear model that is the basis of our mixture model presented in Section 4.3. An important feature of our method is that *any* dynamics model can be used as a black box as long as it provides consistent prediction regions (or, equivalently, variance information). Although our experiments are based on mixture models, our approach is not limited to these models.

Optimal Control: From the perspective of control the main difficulty is the "curse of dimensionality" which makes finding globally optimal solutions impractical. Naive discretization would produce a planning problem of size exponential in the system's dimensionality which is intractable except for small systems. Approximate dynamic programming methods (e.g., [57, 25, 15, 62]) require relevant state features or a distance metric, which tend to be problem-specific and often non-trivial to elucidate. We resort to finding locally optimal policies through nonlinear optimization. Such control methods have been used successfully for some time, e.g., differential dynamic programming [39], iterative LQR [48], and more straight up nonlinear optimization [16, 64]. In this Chapter we only consider shortest path optimal control problems. Our formulation minimizes the trajectory length while imposing that the trajectory reach the goal and that dynamics remain within the prediction region of the learned model. An important feature of our method is that *any* optimal control algorithm can be used as a black box. The optimal control strategy we proposed is just one of the many possible choices.

Exploration: Directing the system to unobserved regions of the state space to collect new information is challenging even in discrete settings. Goal-agnostic exploration methods proposed before, such as those based on entropy, tend to progress slowly, especially in high dimensional systems. Our second contribution is applying the *Optimism-Based Exploration* principle (also known as the *Optimism in the Face of Uncertainty*) to the locally linear continuous setting. The ensuing algorithm works by allowing the planner to pick favorable system dynamics from a prediction region based on the learned model. The amount of optimism allowed is inversely proportional to the amount of data available.

The main advantage of our method is formulating the exploration problem as a simple optimal control problem with *deterministic* dynamics (not necessarily equal to the average dynamics) and additional virtual controls specifying the degree of optimism. Doing so is analogous to the typical reduction employed for discrete systems where the exact exploration problem, a *Partially Observable Markov Decision Process (POMDP)* is reduced (in the *Probably Approximately Correct* sense) to a MDP problem that can be solved efficiently.

The closest competing approach[28] uses Gaussian Processes for modeling and carefully shaped reward functions for driving exploration. Compared to this approach our exploration method has three main advantages. First is simplicity: the competing approach requires solving stochastic optimal control problems while our approach only needs to solve deterministic control problems. Second is modularity: our approach can use any off-the-shelf optimal con-

trol solver and Bayesian dynamics model while the competing approach requires specialized synergy between the planner and the model. Third is continuity: the competing method requires periodically resetting the dynamical system to a fixed start state while our methods works without resets for ergodic systems.

Experiments: We demonstrate our approach on classical reinforcement learning problems: under-actuated pendulum, cartpole and double pendulum swing-up and balance. Additionally we show that the same method can learn to fly a simulated helicopter upside-down (inverted hover) starting from an upright position. Compared to other approaches out method requires less system interaction (is more data-efficient) while making fewer assumptions and relying less on hand-tuned parameters. The competing approach [28] relies on carefully chosen state dependent cost functions that simultaneously serve two purposes: to encourage exploration and to discourage deviation from the target state. Our approach relies on a different mechanism for exploration, namely relaxing dynamics constraints, so we can formulate the same scenarios as shortest path problems. Both in our work and in the competing approach [28] performance is judged in terms of the time required to explore and reach the target state. We optimize this measure of performance directly; the competing approach [28] used a cost function as proxy.

4.2 Bayesian Linear Model

Bayesian linear regression extends regular linear regression, allowing us to also model uncertainty in the parameters by assuming they were drawn from a prior distribution. Let $x(\theta, u)$ be a vector function of the state and controls and let $y(\dot{\theta})$ be an invertible vector function of the time derivative of the state. In the simplest case x might simply be a concatenation of the state and controls and y might simply be equal to the time derivative of the state. This representation allows us the extra flexibility of using different parametrizations for modeling and planning.

We model the stacked vector $z = [y(\dot{\theta}), x(\theta, u)]$ as being drawn from a Gaussian distribution with a Normal-inverse-Wishart prior on the parameters μ, Σ (and we end up with hyperparameters Ψ, ν):

$$\Sigma \sim \mathcal{W}^{-1}(\Psi, \nu), \qquad \mu \,|\, \Sigma \sim \mathcal{N}(m, \Sigma/n), \qquad z \,|\, \mu, \Sigma \sim \mathcal{N}(\mu, \Sigma)$$

The marginal distribution of $x(\theta, u)$ describes the level of accuracy of the model while the conditional distribution of $y(\dot{\theta})$ given $x(\theta, u)$ is identical to the one we would have obtained by Bayesian linear regression and it serves as our dynamics model. Figure 4.1 shows an intuitive example of how conditioning is used for regression. To formally see the connection, we start by partitioning the vector parameter m and the matrix parameter Ψ conformably with y and x. Conditional on x, the predicted y is normally distributed with a Normal-inverse-Wishart

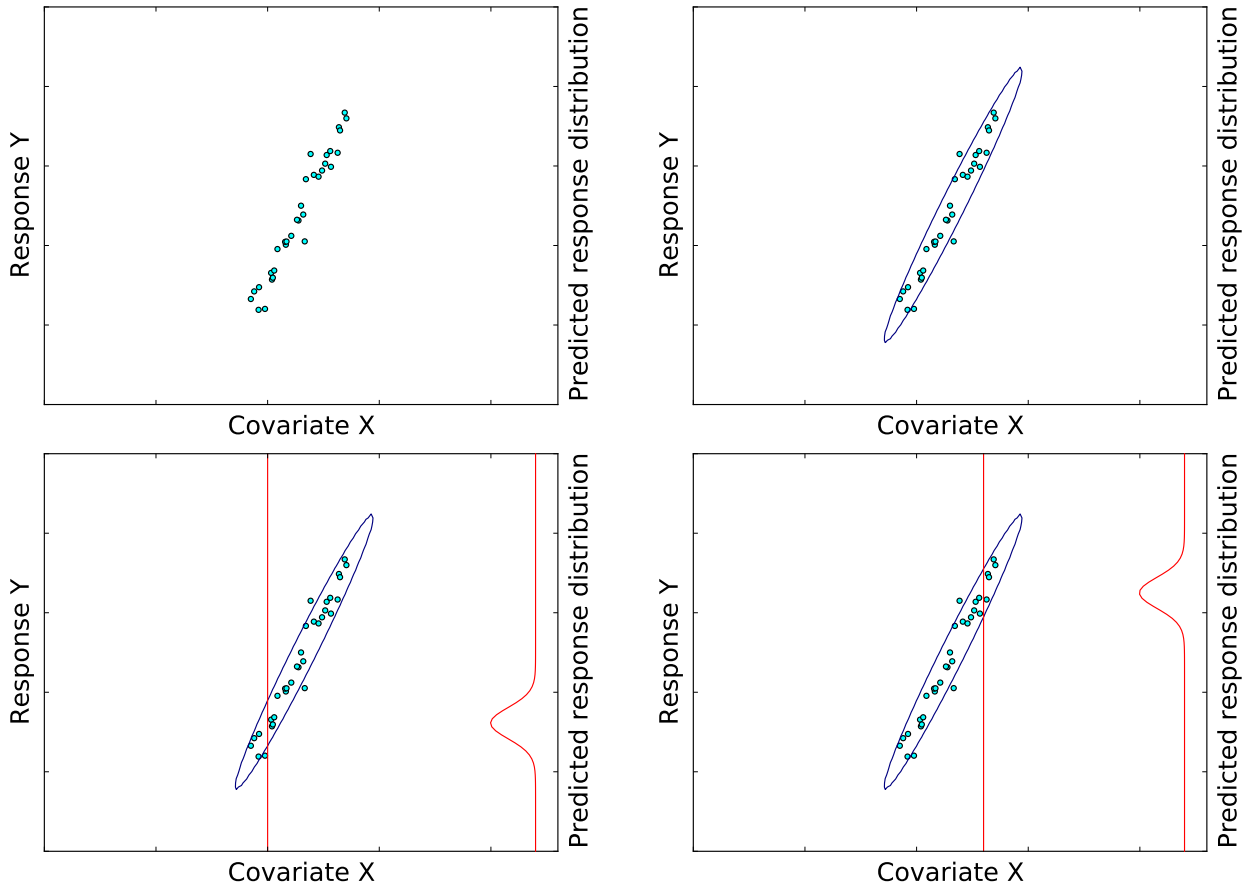


Figure 4.1: Modeling a scalar response Y of a scalar covariate X jointly as a two-dimensional Normal-Inverse-Wishart distribution may be used for Bayesian linear regression via the conditional distribution of Y given X. The blue ellipse show a level set of posterior Normal-Inverse-Wishart distribution. The red line show a "slice" at a given X corresponding to the conditional distribution of Y given X.

prior:

$$\begin{aligned} \Sigma_{\bar{y}\bar{y}} \,|\, x &\sim \mathcal{W}^{-1}(\Psi_{\bar{y}\bar{y}}, \nu) & \Psi_{\bar{y}\bar{y}} &:= \Psi_{yy} - \Psi_{yx}\Psi_{xx}^{-1}\Psi_{xy} \\ \mu_{\bar{y}} \,|\, \Sigma_{\bar{y}\bar{y}}, x &\sim \mathcal{N}(m_y + \Psi_{yx}\Psi_{xx}^{-1}(x - m_x), \Sigma_{\bar{y}\bar{y}}/\bar{n}) & & \\ y \,|\, \mu_{\bar{y}}, \Sigma_{\bar{y}\bar{y}}, x &\sim \mathcal{N}(\mu_{\bar{y}}, \Sigma_{\bar{y}\bar{y}}) & \bar{n} &:= \left(1/n + (x - m_x)^T \Psi_{xx}^{-1}(x - m_x)\right)^{-1}, \end{aligned} \tag{4.1}$$

where the "bar" notation denotes random variables and parameters pertaining to conditional distributions, different from the ones describing the joint distributions.

Expressed in conditional form, the Bayesian Linear Regression Model (4.1) reveals two sources of uncertainty. The distribution on $\mu_{\bar{y}} \,|\, \Sigma_{\bar{y}\bar{y},x}$ captures model *ambiguity* resulting from incomplete knowledge of the system's dynamics. This type of uncertainty can be reduced by further exploration. The distribution on $y \,|\, m_{\bar{y}}, \Sigma_{\bar{y}\bar{y}}, x$ reflects *sensor noise* or non-linearity effects that will not be eliminated by further exploration. Mathematically this becomes clear when we notice that $\Sigma_{\bar{y}\bar{y}}$ will not converge to zero but $\Sigma_{\bar{y}\bar{y}}/\bar{n}$ will converge to zero since $\bar{n}$ grows to infinity as we observe more data.

Therefore the distribution relevant to modeling systems dynamics is determined by $\mu_{\bar{y}} \,|\, x$. Its predictive posterior distribution, obtained by integrating out the prior, is a multivariate t-distribution:

$$\mu_{\bar{y}} \,|\, x \sim \tau_{\nu - d_y + 1}\left(m_y + \Psi_{yx}\Psi_{xx}^{-1}(x - m_x), \frac{\Psi_{\bar{y}\bar{y}}}{\bar{n}(\nu - d_y + 1)}\right) \tag{4.2}$$

where d_y is the dimension of the y vector.

The modelled system dynamics are determined implicitly by the following equation:

$$\begin{aligned} &\left(m_y - \Psi_{yx}\Psi_{xx}^{-1}m_x + \Psi_{yx}\Psi_{xx}^{-1}x(\theta, u) - y(\dot{\theta})\right) - \left(\frac{\Psi_{\bar{y}\bar{y}}}{\bar{n}(\theta, u)(\nu - d_y + 1)}\right)^{\frac{1}{2}} \omega = 0, \\ &\text{written equivalently as} \quad \hat{f}(\dot{\theta}, \theta, u, \omega) = 0, \end{aligned} \tag{4.3}$$

where ω is a vector of d_y independent and identically distributed student's t-distributed random variables with unit scale parameter and $\nu - d_y + 1$ degrees of freedom. Asymptotically, after seeing a large number of observations $\bar{n}(\theta, u)$ converges to n and ω converges in distribution to a vector of d_y independent and identically distributed standard normal random variables.

4.3 Dirichlet Process Mixture of Linear Models

A single Gaussian distribution (or, equivalently, a single linear model) is not sufficient to capture the dynamics of complex nonlinear systems, but a mixture of Gaussians is often descriptive enough if we assume that, for each sufficiently small region in state space, the underlying (unknown) dynamical system is locally linear. Unfortunately, it is generally not

possible to determine the optimal number of clusters a priori and this parameter has a large impact—too many clusters leads to over-fitting and too few clusters collapses modes that may be possible to discriminate when we have large amounts of data. Dirichlet process mixture models (DPMMs) tackle this issue by allowing a countably infinite number of clusters, with the actual number utilized for a given data set determined via posterior inference. Under the Dirichlet process prior the expected number of clusters grows logarithmically in the number of observations [63] and posterior inference sharpens this growth rate so as to control model complexity as we observe more data.

4.3.1 The Standard Dirichlet Process Mixture Model.

Our model is based on the following standard DPMM [17]:

$$G \,|\, \alpha, \lambda \sim \mathrm{DP}(\alpha, \mathcal{NW}^{-1}(\lambda)), \qquad \eta_n \,|\, G \sim G, \qquad z_n \,|\, \eta_n \sim \mathcal{N}(\eta_n),$$

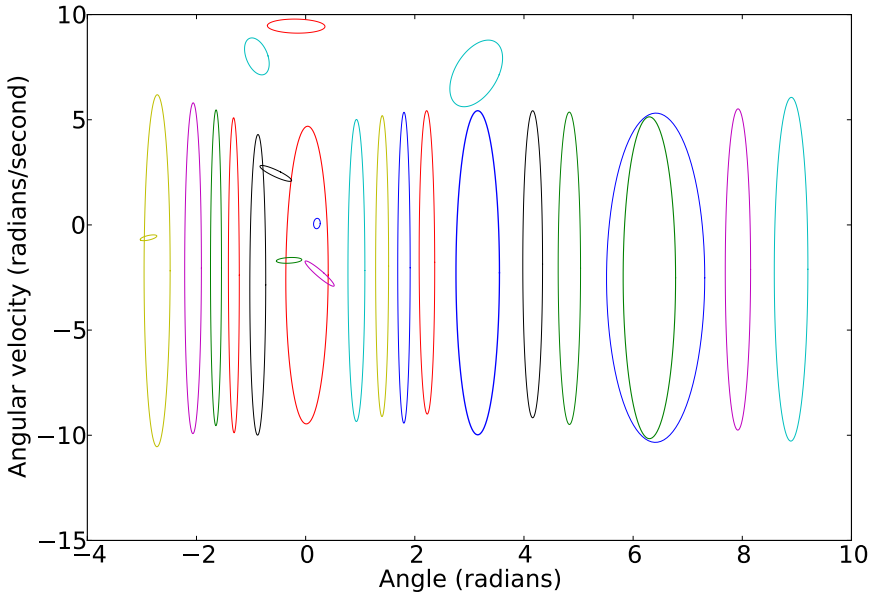
where $\mathcal{N}$ is the Gaussian distribution, where α is the concentration parameter of the Dirichlet process prior and where the base measure is the Normal-inverse-Wishart distribution with hyperparameter λ. As done previously in Section 4.2, let $x(\theta, u)$ be a vector function of the state and controls, and let $y(\dot{\theta})$ be an invertible vector function of the time derivative of the state. We are modeling the stacked vector $z = [y(\dot{\theta}), x(\theta, u)]$. Given the importance of α in DPMMs, we treat it not as a fixed hyperparameter but endow it with a gamma prior distribution, $\mathrm{Ga}(0, 0)$, which we integrate over via the posterior inference algorithm.

It is convenient to rewrite this model using an exponential family representation. The natural parameter vector for the Normal-inverse-Wishart prior is obtained by stacking together the usual parameters: $\lambda = [n\mu, \Psi + n\mu\mu^T, n, \nu + 2 + d_z]$, where d_z is the dimension of z. The vector of sufficient statistics is also obtained by concatenation: $T(z) = [z, zz^T, 1, 1]$. What makes this representation convenient is that posterior updates are now extremely simple: $\tau = \lambda + T(z)$.

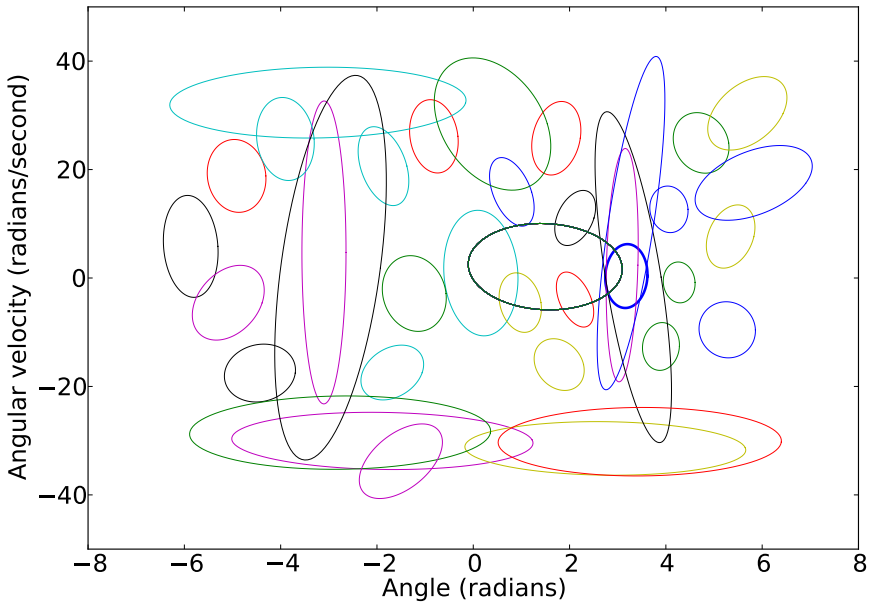
4.3.2 Variational Inference.

While Dirichlet process mixture models are often fit with one of a number of different Markov chain Monte Carlo or sequential Monte Carlo algorithms, given our stringent requirements for computational efficiency we have chosen instead to make use of variational inference methods. In particular we make use of the mean-field variational inference procedure for DPMMs developed by Blei and Jordan [17]. This procedure, based on a stick-breaking representation of the Dirichlet process, requires us to set an upper bound for the number of clusters that can be represented. This is a parameter of the inference procedure, not of the model, and it is generally set to be as high as computational resources permit. Many of the represented clusters will not be populated.

The mean-field variational inference algorithm of Blei and Jordan [17] can be viewed as an approximate version of *Expectation-Maximization* that iterates between the following two



(a) Under-actuated pendulum. The maximum number of clusters was set to 100 and the prior weight, w, was set to 10^{-2}. We simulated a single trajectory, 200 seconds long, starting with the pole resting downwards and changing the control randomly at 2Hz.



(b) Cartpole. The maximum number of clusters was set to 100 and the prior weight, w, was set to 1. We simulated 20 trajectories, each 5 seconds long, starting with the pole resting downwards and changing the control randomly at 10Hz.

Figure 4.2: Clusters learned by batch variational inference on the trajectories of two dynamical systems under random controls. Both systems are described in Section 4.6. Extra dimensions corresponding to accelerations and control values are not shown. The different cluster patters reflect the different structures of non-linearities.

steps:

$$\text{E step:} \quad \varphi_{k,n} \propto \exp\left(\nabla A(\tau_k)^T \cdot T(z_n) + \psi(a_k) - \psi(b_k) + \sum_{i=1}^{k}(\psi(b_i) - \psi(a_i + b_i))\right) \tag{4.4}$$

$$\text{M step:} \quad \tau_k = \lambda + \sum_n \varphi_{k,n} T(z_n), \quad a_k = 1 + \sum_n \varphi_{k,n}, \quad b_k = \alpha + \sum_n \sum_{j=k+1}^{K} \varphi_{j,n} \tag{4.5}$$

where ψ is the digamma function, where A is the partition function of the Normal-inverse-Wishart prior, and where $\varphi_{k,n}$ and τ_k are variational parameters that are the degrees of freedom of the algorithm. To initialize the base measure we start with $\lambda_0 = [0, 0, 0, 2d_z + 1]$ and we perform a weighted posterior update imagining that the entire training set belongs to a single cluster:

$$\lambda = \lambda_0 + w \cdot \sum_n T(z_i) / \sum_n T(z_i)_{-1}$$

where $T(x_i)_{-1}$ is the last component of the vector of sufficient statistics and w specifies the relative importance of the prior with respect to a single observation; we set it to .1. With this choice of prior parameters the clustering method is invariant to affine transformations of the dataset. Figure 4.2 shows the clusters learned by this method on trajectories of dynamical systems.

4.3.3 Dynamics Prediction.

Once the model has been trained, we would like to use it to make predictions of of $y(\dot{\theta})$, conditional on $x(\theta, u)$. Figure 4.3 shows a simple example of how these predictions may be used for regression. A difficulty in the DPMM setting is that it is possible for the conditional distribution under the model to be multi-modal, which can complicate planning by introducing local optima. It also creates difficulties in setting prediction regions that are required for our exploration method (described in Section 4.5). To address these difficulties we use the approximation methodology discussed below.

Let $p(k \,|\, x)$ be the posterior cluster assignments computed from the trained model by using Bayes' theorem:

$$p(k \,|\, x) = \frac{p(x \,|\, \tau_k)p(k)}{\sum_l p(x \,|\, \tau_l)p(l)}, \quad p(k) := \log(a_k) - \log(a_k + b_k) + \sum_{l=k+1}^{K} (\log(b_l) - \log(a_l + b_l))$$

The correct conditional density is given by $p(y \,|\, x) = \sum_k p(k \,|\, x)p(y \,|\, x, \tau_k)$. Our approximation is $p(y \,|\, x, \tau_x)$ where $\tau_x = \sum_k p(k \,|\, x)\tau_k$, which is justified by its connection to weighted

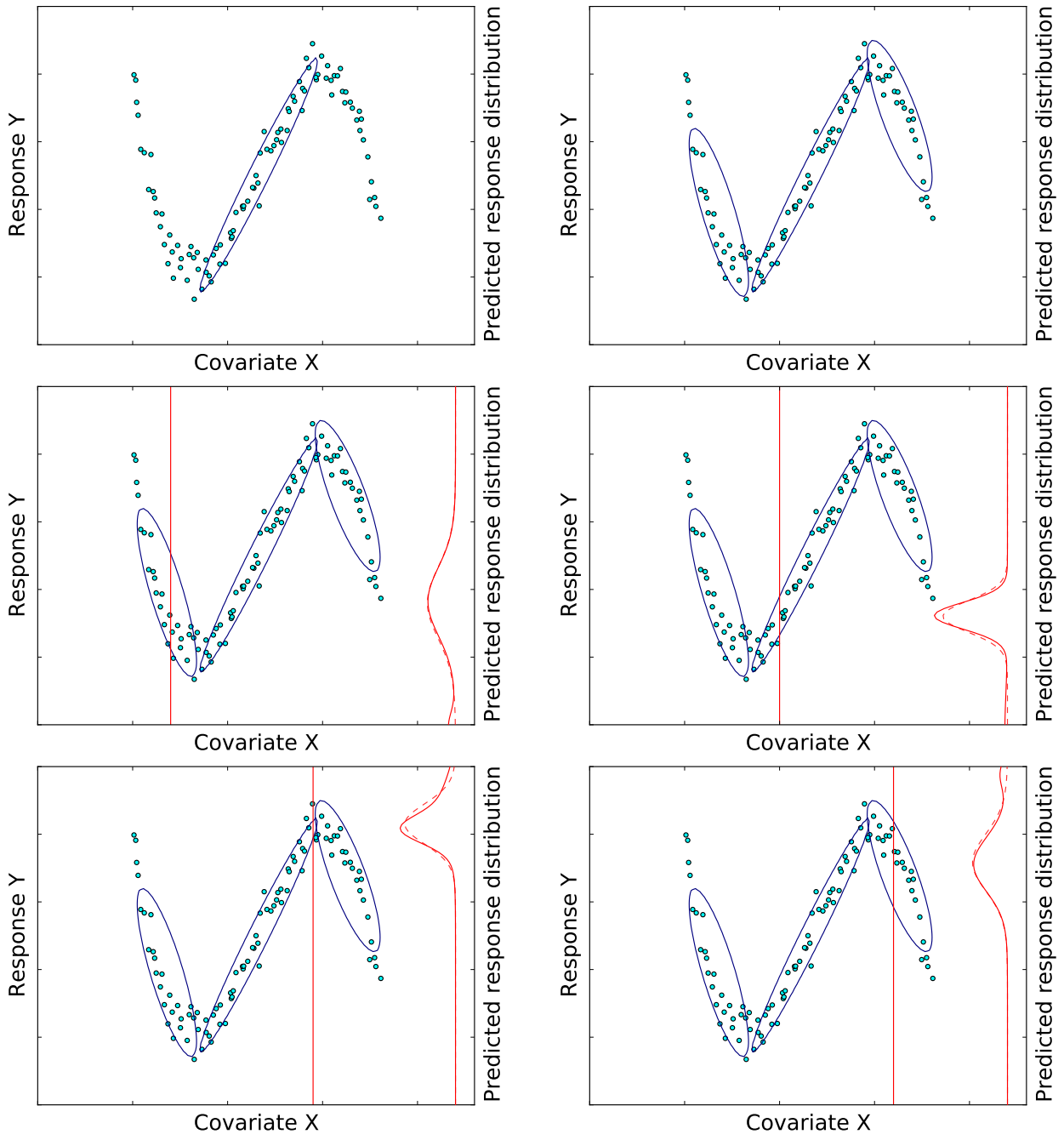


Figure 4.3: Modeling a scalar response Y of a scalar covariate X jointly as a mixture model may be used for Bayesian linear regression via the conditional distribution of Y given X. The blue ellipses show level sets of the Normal-Inverse-Wishart clusters. The red line show a "slice" at a given X corresponding to the conditional distribution of Y given X. The dotted red line shows the closest Normal-Inverse-Wishart approximation in KL divergence.

least squares. We can now substitute the expression of τ_k in terms of the sufficient statistics of the training data:

$$\tau_x = \lambda + \sum_{k,n} p(k \mid x)\varphi_{k,n}T(z_n), \tag{4.6}$$

where $\varphi_{k,n}$ is prescribed by the Equation (4.4). The resulting approximate prediction is the same as the one we would obtain by weighted Bayesian Linear Regression with factored weights $v_n = \sum_k p(k \mid x)\varphi_{k,n}$. Average predictions under this model correspond to predictions made by weighted least squares with these weights. Weighted least squares is generally considered robust and accurate but computationally expensive[61] because normally the whole training set needs to be accessed before making each prediction. The key for the viability of our approach is that the Dirichlet process mixture model can be seen as a form of *data compression* that makes weighted least squares computationally tractable.

Since our approximate predictive density is simply a Normal-Inverse-Wishart (NIW) distribution we use Equation (4.3) to formulate a prediction region. Comparing the weighted least squares approximation discussed above and the simple linear model discussed in Section 4.2, also based on NIW distribution, we see that the key difference is the additional dependence on $x(\theta, u)$ of the distribution parameters Ψ, μ, ν, n summarized by τ_x through the predictive cluster responsibilities $p(k \mid x)$. There is no such dependence in the basic linear model. The following equations summarize our weighted least squares approximation:

$$\begin{aligned}
[n\mu, \Psi + n\mu\mu^T, n, \nu + 2 + d_z] := \tau_x &= \lambda + \sum_{k,n} p(k \mid x)\varphi_{k,n}T(z_n) \\
\Psi_{\bar{y}\bar{y}} &:= \Psi_{yy} - \Psi_{yx}\Psi_{xx}^{-1}\Psi_{xy} \\
\bar{n} &:= \left(1/n + (x - m_x)^T\Psi_{xx}^{-1}(x - m_x)\right)^{-1},
\end{aligned}$$

and the prediction region is defined by:

$$\left(m_y - \Psi_{yx}\Psi_{xx}^{-1}m_x + \Psi_{yx}\Psi_{xx}^{-1}x(\theta, u) - y(\dot{\theta})\right) - \left(\frac{\Psi_{\bar{y}\bar{y}}}{\bar{n}(\theta, u)(\nu - d_y + 1)}\right)^{\frac{1}{2}}\omega = 0, \tag{4.7}$$

written equivalently as $\hat{f}(\dot{\theta}, \theta, u, \omega) = 0$.

4.4 Optimal Control

In the spirit of past work on nonlinear optimization for finding locally optimal control policies [16, 64, 23, 54] we find an open-loop sequence of control inputs, and then close the loop by continuously re-planning during execution instead of explicitly model policies. Many exploration methods require re-planning anyway [10], so computing entire policies seems wasteful. This control approach is becoming more and more prevalent as computers have become fast enough.

4.4.1 Continuous Time Formulation

Let $\theta_t \in \mathbb{R}^{d_\theta}$ and $u_t \in \mathbb{R}^{d_u}$ be the state of the system and the control applied at time t. The time derivative of the state is $\dot{\theta}_t \in \mathbb{R}^{d_\theta}$. Dynamics are determined implicitly by the set of equations $f(\dot{\theta}, \theta, u) = 0 \in R^{d_\theta}$ where we call f the *implicit dynamics function*. For example, in the case of a linear system the implicit dynamics function is $f(\dot{\theta}, \theta, u) = A\theta + Bu - \dot{\theta}$.

We are only considering shortest path optimal control problems in this Chapter. Solving these problems is equivalent to directly optimizing the performance metric reported by the competing approach [28] (their cost function is different from the performance metric). The objective is finding a sequence of states and controls that takes the system from a specified start state to a (potentially partially) specified goal state in the shortest amount of time. Let h be the time horizon. The shortest path problem can be formulated as the constrained functional optimization problem below:

$$\begin{aligned} \text{minimize}_{\ \theta,u,h}: \quad & h & & (4.8) \\ \text{subject to}: \quad & f\left(\dot{\theta}(t), \theta(t), u(t)\right) = 0 & \forall\ t \in [0, h] \\ & -1 \leq u(t) \leq 1 & \forall\ t \in [0, h] \\ & \theta(0) = \theta_{\text{start}}, \quad \theta(h) = \theta_{\text{goal}}, \quad h > 0 \end{aligned}$$

4.4.2 Non-Linear Program Formulation

Time discretization is the first step towards addressing the optimization problem (4.8) in practice. Instead of imposing the dynamics constraints at all times, we now only impose them at a finite number (k) of appropriately chosen *collocation times* $0 \leq h\tau_1 \leq h\tau_2 \leq \cdots \leq h\tau_k \leq h$. For example, in the simplest case of Euler integration we would choose $\tau_i = (i-1)/(k-1)$. To simplify notation, let $\dot{\theta}_i = \dot{\theta}(h\tau_i)$.

We eliminate the states $\theta(t)$ from the optimization problem since they can be expressed as time integrals of the time derivatives of states as follows:

$$\theta(h\tau_i) - \theta(0) = h \int_0^{\tau_i} \dot{\theta}(h\tau) d\tau \approx h \sum_j A_{ij}(k) \dot{\theta}_j$$

$$\theta(h) - \theta(0) = h \int_0^{1} \dot{\theta}(h\tau) d\tau \approx h \sum_j w_j \dot{\theta}_j$$

where the approximate integration operators A_{ij} and w_j are chosen appropriately depending on the type of time discretization scheme used. For example, in the simplest case of Euler integration we would choose $A_{ij} = \delta_{i \leq j}/k$ and $w_j = 1/k$. The optimal control problem can now be expressed as a discrete time non-linear optimization problem:

$$\begin{aligned} \text{minimize}_{\ \dot{\theta}_i, u_i, h}: \quad & h & (4.9)\\ \text{subject to}: \quad & f\left(\dot{\theta}_i, \theta_{\text{start}} + h\sum_j A_{ij}\dot{\theta}_j, u_i\right) = 0 \qquad \forall\ i \in 1, 2, \cdots, k \\ & -1 \le u_i \le 1 \qquad \forall\ i \in 1, 2, \cdots, k \\ & \theta_{\text{goal}} - \theta_{\text{start}} = h\sum_j w_j\dot{\theta}_j, \quad h > 0 & (4.10) \end{aligned}$$

Pseudospectral methods [58, 14] specify efficient time discretization schemes grounded on implicit integration theory. Time steps $\tau_1, \ldots \tau_k$ are not equally spaced; they are more densely clustered around the endpoints. The approximate integration operators A_{ij} and w_j are prescribed accordingly. Advantages of pseudospectral methods include:

- Fewer collocation times can be used without compromising accuracy.
- The duals variables of the time-discretized optimization problem are provably close to the co-states of the continuous time optimization problem.
- Interpolation of the resulting controls between collocation time steps is well specified.

4.4.3 Solving the Non-Linear Program

The optimization problem (4.9) is non-convex so in general we can only guarantee convergence to a (potentially infeasible) local optimum. In our experiments we used a specialized Sequential Linear Programming solver [56] discussed below but any other efficient NLP solver may be used instead.

The change of variables to $h\dot{\theta}_i, u_i, 1/h$ helped alleviate the local optima issue in our experience since final constraint given by Equation (4.10) becomes linear. For simplicity of notation let all the decision variables be stacked into a single vector $\xi = [1/h, h\theta_1, u_1, h\theta_2, u_2, \ldots, h\theta_k, u_k]$ and define linear operators H, T_i, U_i such that $H\xi = 1/h$, $T_i\xi = h\theta_i$ and $U_i\xi = u_i$. With this notation, the non-linear program (4.9) can be written equivalently as:

$$\begin{aligned} \text{minimize}_{\ \xi}: \quad & -H\xi \\ \text{subject to}: \quad & f_i(\xi) = 0, \qquad -1 \le U_i\xi \le 1 \qquad \forall\ i \in 1, \cdots, k \\ & \theta_{\text{goal}} - \theta_{\text{start}} = \sum_j w_j \cdot T_j\xi, \quad H\xi > 0 \\ \text{where} \quad & f_i(\xi) := f\left(H\xi \cdot T_i\xi, \theta_{\text{start}} + \sum_j A_{ij} \cdot H_j\xi, U_i\xi\right) \end{aligned}$$

We solve the NLP by iterating the following three steps until convergence starting with an appropriate initialization ξ.

Step 1: Check feasibility of the *optimality first order approximation* (OFOA), a linear program that includes the original objective:

$$\begin{aligned} \text{minimize}_{\ \delta\xi}: \quad & -H(\xi+\delta\xi) && \text{OFOA} \\ \text{subject to}: \quad & f_i(\xi)+\frac{\partial f_i}{\partial \xi}\cdot\delta\xi=0, \quad -1\le U_i(\xi+\delta\xi)\le 1 && \forall\ i\in 1,\cdots,k \\ & \theta_{\text{goal}}-\theta_{\text{start}}=\sum_j w_j\cdot T_j(\xi+\delta\xi), \quad H\xi>0 \end{aligned} \tag{4.11}$$

Step 2: If there is no solution to OFOA above then formulate and solve the *feasibility first order approximation* (FFOA) shown below where the objective is to locally reduce the L1 penalty on the constraints. This linear program is always feasible.

$$\begin{aligned} \text{minimize}_{\ \delta\xi}: \quad & \sum_i w_i\left\|f_i(\xi)+\frac{\partial f_i}{\partial \xi}\cdot\delta\xi\right\|_1 && \text{FFOA} \\ \text{subject to}: \quad & -1\le U_i(\xi+\delta\xi)\le 1 && \forall\ i\in 1,\cdots,k \\ & \theta_{\text{goal}}-\theta_{\text{start}}=\sum_j w_j\cdot T_j(\xi+\delta\xi), \quad H\xi>0 \end{aligned} \tag{4.12}$$

Step 3: Perform a line search along the direction specified by the solution $\delta\xi$ to the OFOA if feasible or to the FFOA otherwise. The objective is to minimize the feasibility error measured in L1 norm:

$$\xi \leftarrow \underset{0\le\alpha\le 1}{\operatorname{argmin}} \sum_i w_i\|f_i(\xi+\alpha\cdot\delta\xi)\|_1. \tag{4.13}$$

Return to Step 1 unless the change in ξ is small.

4.5 Optimism-Based Exploration

Optimism-Based Exploration (also known as the *Optimism in the Face of Uncertainty*) is a general principle for model-based reinforcement learning underlying numerous efficient algorithms backed by theoretical guarantees in different settings:

- Multi-Armed bandit problems where *Upper Confidence Based (UCB)* policies have proven successful [5, 13, 12, 11, 34].
- Discrete *Markov Decision Processes*, e.g. the *Bayesian Optimistic Local Transitions* algorithm [8] or the *Optimistic Linear Programming* algorithm [7].
- Linear systems with quadratic costs [3].

Algorithm 3 Exploration algorithm

Require: dynamical system, goal state θ_{goal}
Require: sampling frequency ν_s, control frequency ν_c
  $\pi \leftarrow$ zero control policy
  $\theta_{\text{start}} \leftarrow$ random initial state for dynamical system
  $z \leftarrow$ empty list of regression features
  repeat
    $\theta_{\text{start}} \leftarrow$ final state after executing policy π for $1/\nu_c$ seconds starting at state θ_{start}
    $z \leftarrow$ appended to z regression features $z_t = [y(\dot{\theta}_t, x(\theta_t, u_t))]$ sampled every $1/\nu_s$ seconds
    $\phi \leftarrow$ randomly initialized cluster responsibilities
    repeat
      $\tau \leftarrow$ M step per Equation 4.5 given ϕ and sufficient statistics $T(z)$
      $\phi \leftarrow$ E step per Equation 4.4 given τ and sufficient statistics $T(z)$
    until ϕ converges
    $\hat{f}(\dot{\theta}, \theta, u, \omega) \leftarrow$ learned dynamics per Equation 4.7 given ϕ and sufficient statistics $T(z)$
    $\xi \leftarrow$ optimal control decision variables initialized
    repeat
      $\delta\xi \leftarrow$ solve the OFOA per Equation 4.11 given dynamics $\hat{f}$ with controls u and ω.
      if the OFOA is infeasible **then**
        $\delta\xi \leftarrow$ solve the FFOA per Equation 4.12 given dynamics $\hat{f}$ with controls u and ω.
      end if
      $\xi \leftarrow$ line search per Equation 4.13 given dynamics $\hat{f}$ and $\delta\xi$.
    until ξ converges
    $\pi \leftarrow$ open loop control policy with control sequence u_i extracted from ξ
  until θ_{start} stabilizes at θ_{goal}

Optimism-based exploration relies on allowing the planning agent to be optimistic with respect to the learned dynamics. In addition to being able to pick actions, the agent is now also allowed to pick favorable dynamics f from a prediction region $\mathcal{F}$. Deciding which dynamics are favorable depends on the planning goal so exploration prioritizes learning features of the dynamics that are relevant to reaching the goal. The prediction region $\mathcal{F}$ is provided by the dynamics model and it shrinks as more data is observed, eventually forcing the agent to plan according to the average dynamics.

One of our contributions is applying the optimism-based exploration principle to non-linear dynamical systems. In this section we will show how applying this principle allows us to formulate the exploration problem as a simple optimal control problem with *deterministic* dynamics but additional virtual controls. Doing so allows us to leverage the optimal control approach discussed in Section 4.4, or any other black-box optimal control solver that might be available. If the prediction region $\mathcal{F}$ is provided explicitly then optimism-based exploration can be formulated as the following continuous time optimal control problem obtained by

adding f as a optimization variable to our previous formulation in Equation (4.8):

$$\begin{aligned} \text{minimize}_{\ f,\theta,u,h}: \quad & h & \\ \text{subject to}: \quad & f_t\left(\dot{\theta}(t), \theta(t), u(t)\right) = 0 & \forall\ t \in [0,h] \\ & f_t \in \mathcal{F}, \qquad -1 \leq u(t) \leq 1 & \forall\ t \in [0,h] \\ & \theta(0) = \theta_{\text{start}}, \quad \theta(h) = \theta_{\text{goal}}, \quad h > 0 & \end{aligned} \tag{4.14}$$

Note that the planning agent is allowed to chose different favorable dynamics f_t at different time steps; this consistent with the formulation of previously proposed algorithms based on the same principle [5, 13, 12, 11, 34, 8, 7, 3].

The models discussed in Sections 4.2 and 4.3 allow us to parameterize the prediction region $\mathcal{F}$ in terms of the Bayesian model uncertainty ω. We define $\mathcal{F} = \{f(\dot{\theta}, \theta, u) = \hat{f}(\dot{\theta}, \theta, u, \omega)$ for all ω such that: $-1 \leq \omega \leq 1\}$ where $\hat{f}$ represents the Bayesian dynamics model as shown by Equation (4.3) in the case of a linear model. The multi-dimensional prediction region thus defined is centered around the expected dynamics and extends one standard deviation in each direction. Once the prediction region has been parametrized, we can re-write the exploration optimal control problem as an instance of a generic optimal control problem defined by Equation (4.8) which, in addition to the physical controls u, now has a number of virtual controls ω:

$$\begin{aligned} \text{minimize}_{\ \theta,u,\omega,h}: \quad & h & \\ \text{subject to}: \quad & \hat{f}\left(\dot{\theta}(t), \theta(t), u(t), \omega(t)\right) = 0 & \forall\ t \in [0,h] \\ & -1 \leq \omega(t) \leq 1, \qquad -1 \leq u(t) \leq 1 & \forall\ t \in [0,h] \\ & \theta(0) = \theta_{\text{start}}, \quad \theta(h) = \theta_{\text{goal}}, \quad h > 0 & \end{aligned} \tag{4.15}$$

Larger or smaller prediction regions $\mathcal{F}$ may be used as necessary to increase or reduce the amount of exploration respectively. Prediction regions of different sizes may be defined by replacing the constraint $-1 \leq \omega \leq 1$ with $-p \leq \omega \leq p$ for some fixed $p > 0$ which now allows p standard deviations from the mean in each direction. We found that resizing the prediction region was not necessary in our experiments and we set $p = 1$ accordingly.

Algorithm 3 summarizes our full algorithm including modeling, control and exploration.

4.6 Experiments

The experiments in this section are meant to show the generality of our approach; we have not optimized the algorithm for any specific task. An important feature of our approach is that there are only a few scalar parameters to tune. The clustering prior weight w and the prediction region parameter p which governs exploration were set to values that seemed reasonable a-priori ($w = .1, p = 1.0$). Additional parameters, the maximum number of

Table 4.1: Summary of experimental results showing the total interaction time required to learn an appropriate model and solve each task. See Section 4.6 for details.

	Pendulum	Cartpole	Double Pendulum	Helicopter
Our method	4 ± 1 s	10 ± 4 s	18 ± 8 s	4.5 ± 1.5 s
Best competing	$15 - 60$ s	17.5 s	50 s	$30 - 60$ min

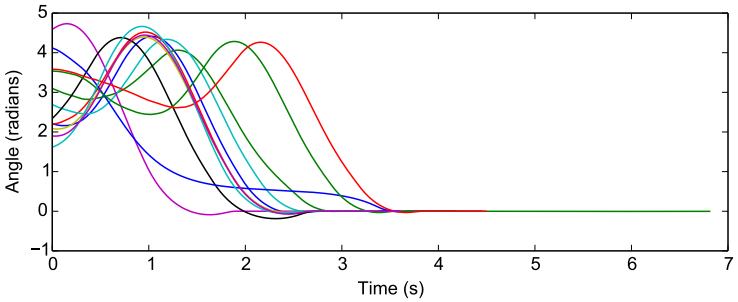
clusters that may be used by the mixture model and the number of collocation times for optimal control should be set as high as possible to increase accuracy. In our experiments we set these parameters to 100 and 25 respectively. We recorded state and control observations at 100 Hz and we re-planned at 20Hz with exploration. Performance can be further improved with specific parameter tuning.

Comparing our non-linear program solver presented in Section 4.4.3 to the off-the-shelf solver Knitro (version 8.1) showed that both typically converge to the same solutions. In terms of number of iterations needes for convergence, our solver typically out-performs Knitro but not always. Using Gaussian pseudospectral [14] time discretization lead to the same solution accuracy as Euler discretization but required between 5 times to 10 times fewer collocation times, resulting in significant speedups. In our experiments we used 35 collocation time steps.

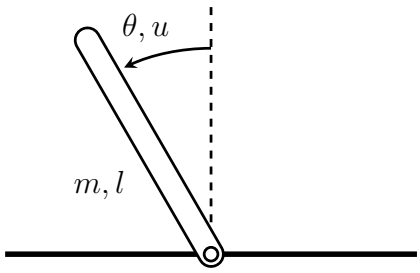
The work of Deisenroth et al. [30, 29, 28] based on Gaussian Processes is closest in spirit to ours. As shown by the experiments discussed below, our method requires less time to learn appropriate dynamics models and solve tasks also addressed by the competing approach: under-actuated pendulum, cartpole, and double pendulum swing-up and balance. The most significant difference in methodology is the exploration technique. They rely on implicit exploration guided by the reward function while we address the exploration-exploitation trade-off directly. In other words, they view uncertainty as noise and plan policies that are robust to it. Our approach considers remaining model uncertainty as opportunity for exploration. Additionally, their method requires known reward functions, the ability to reset the dynamical systems, and parametric forms for policies; our method requires neither. We judge performance as done in the competing approach [28], in terms of the total time (including exploration) required to reach the target state. In our approach this performance metric is optimized directly; in the competing approach [28] it is optimized indirectly via a carefully chosen cost function that used both to encourage exploration and to discourage deviation from the target state.

4.6.1 Under-Actuated Pendulum Swing-Up and Balance.

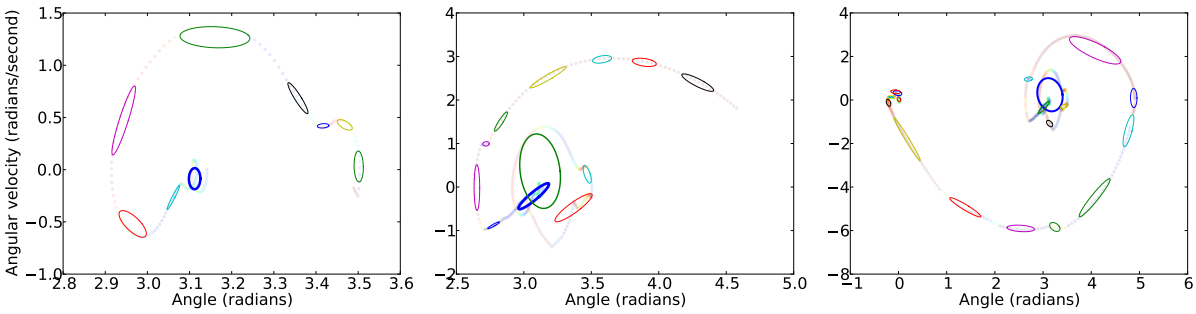
We simulated an under-actuated pendulum system where the dynamics and choice of parameters matches that of Deisenroth et al. [30]. The goal is to swing the pendulum up from the initial state hanging down to the upright balanced state. One swing is necessary to provide



(a) Lines show different independent trials that are halted once the system remains stable about the target state for one second.



(b) The torque that can be applied to the pendulum is constrained.



(c) Clusters learned by Dirichlet Process mixture model at different stages of exploration

Figure 4.4: Pendulum swing-up and balance experiments. The goal is to swing the pendulum up from the initial state hanging down to the upright balanced state. One swing is necessary to provide enough momentum due to torque constraints.

enough momentum due to torque constraints.

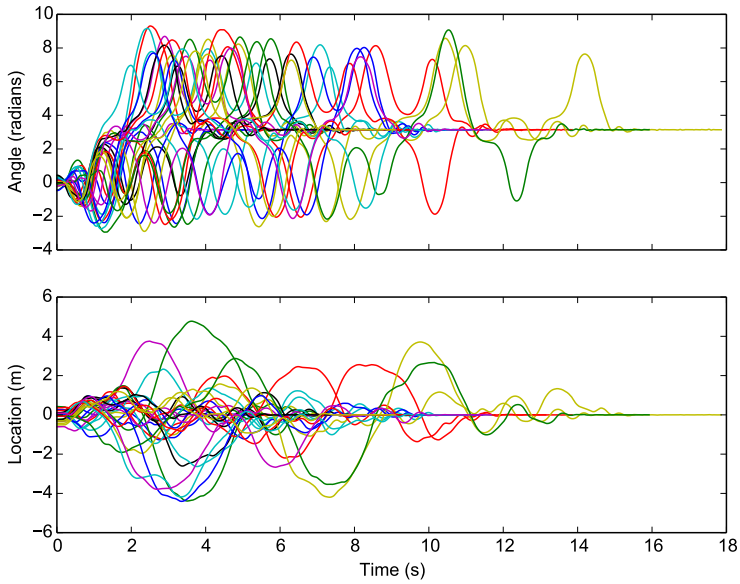
The equation of motion for the pendulum is:

$$\ddot{\theta}ml^2 - ku + b\dot{\theta} - mgl\sin(\theta) = 0$$

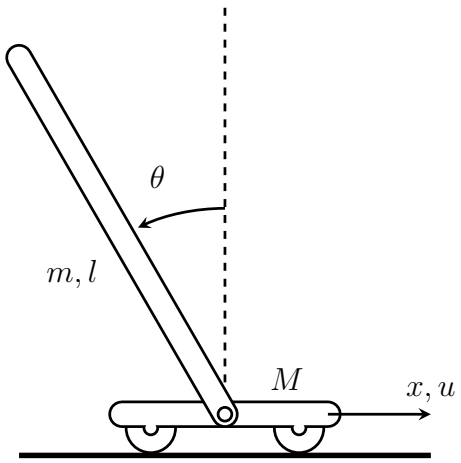
with parameters:

$$\begin{aligned} l &= 1.0, && \text{length of pendulum } (m) \\ m &= 1.0, && \text{mass of pendulum } (kg) \\ b &= 0.05, && \text{coefficient of friction} \\ g &= 9.82, && \text{acceleration of gravity } (m/s^2) \\ k &= 5.0, && \text{maximum torque } (Nm) \end{aligned}$$

The start state $[\dot{\theta}, \theta]$ is sampled from a normal distribution with mean $[0, \pi]$ and diagonal variance $[1, 1]$.



(a) Lines show different independent trials that are halted once the system remains stable about the target state for one second.



(b) Cartpole system diagram. The cart wheels are actuated but the pole is not.

Figure 4.5: Cartpole swing-up and balance experiments. The goal is to swing the pendulum up from the initial state hanging down to the upright balanced state.

We modeled the system using a Dirichlet Process Mixture of Linear Models as described in Section 4.3. The model is set up to predict the angular acceleration of the pole given the sine and cosine of the pole angle, the angular velocity and the control applied. Figure 4.4 shows a few runs of our algorithm. Swing-up was successful in every run after 4 ± 1 seconds of interaction. The competing Gaussian Process based approach[30] requires between 15 - 60 seconds for the same task.

4.6.2 Cartpole Swing-Up and Balance.

We simulated a classical cart-pole system with the dynamics and parameters used by the state-of-the art method based on Gaussian Processes [28]. The goal is to swing-up the pole from its inert state hanging down to the upright balanced state.

The equations of motion for cartpole are:

$$\begin{aligned} 0 = &- \ddot{\theta} l(4M + 4m - 3m\cos(\theta)^2) - 3mlw^2 \sin(\theta)\cos(\theta) - \\ &- 6(M+m)g\sin(\theta) - 6(ku - b\dot{x})\cos(\theta) \\ 0 = &- \ddot{x}(4M + 4m - 3m\cos(\theta)^2) + 2mlw^2\sin(\theta) + 3mg\sin(\theta)\cos(\theta) + 4ku - 4b\dot{x} \end{aligned}$$

with parameters:

$$\begin{aligned} l &= 0.5, && \text{length of pendulum (m)} \\ m &= 0.5, && \text{mass of pendulum (kg)} \\ M &= 0.5, && \text{mass of cart (kg)} \\ b &= 0.1, && \text{coefficient of friction between cart and ground} \\ g &= 9.82, && \text{acceleration of gravity (m/s}^2\text{)} \\ k &= 10.0, && \text{maximum control (N)} \end{aligned}$$

The start state $[\dot{\theta}, \dot{x}, \theta, x]$ is sampled from a normal distribution with mean $[0, 0, 0, 0]$ and diagonal variance $[0, 0, .5, .5]$.

We modeled the system using a Dirichlet Process Mixture of Linear Models as described in Section 4.3. The model is set up to predict the acceleration of the cart and the angular acceleration of the pole given the sine and cosine of the pole angle, its angular velocity and the control applied. Figure 4.5 shows a diagram of the system and a few different runs of our algorithm. It successfully learned the required model and solved the task after 10 ± 4 seconds of interaction and succeeded in all runs. The competing Gaussian Process based approach [28, 29] required a typical exploration time of 17.5 and has a 95% success rate.

4.6.3 Double Pendulum Swing-Up and Balance.

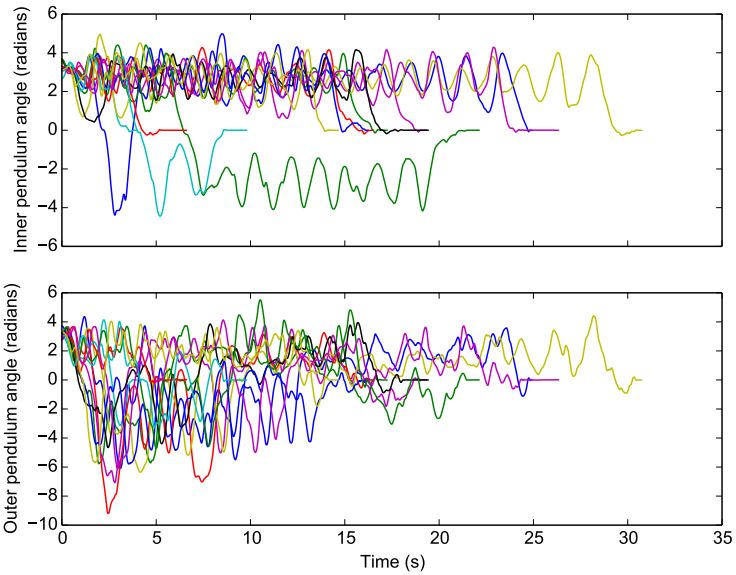
We simulated a double-pendulum system with the dynamics and parameters used by the state-of-the art method based on Gaussian Processes [28]. The goal is to swing-up both poles from the inert state hanging down to the upright balanced state.

The equations of motion for the double pendulum are given by:

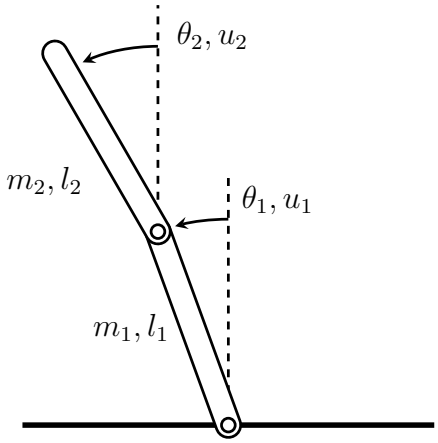
$$\begin{aligned} 0 = &\begin{bmatrix} l_1^2(m_1/4 + m_2) + I_1 & m_2 l_1 l_2 \cos(\theta_1 - \theta_2))/2 \\ m_2 l_1 l_2 \cos(\theta_1 - \theta_2)/2 & l_2^2 m_2/4 + I_2 \end{bmatrix} \cdot \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} - \\ &- \begin{bmatrix} g l_1 \sin(\theta_1)(m_1/2 + m_2) - m_2 l_1 l_2 \dot{\theta}_2^2 \sin(\theta_1 - \theta_2)/2 + k u_1 \\ m_2 l_2 (l_1 \theta_1^2 \sin(\theta_1 - \theta_2) + g \sin(\theta_2))/2 + k u_2 \end{bmatrix} \end{aligned}$$

with parameters:

$$\begin{aligned} m_1 &= 0.5, && \text{mass of 1st link } (kg) \\ m_2 &= 0.5, && \text{mass of 2nd link } (kg) \\ l_1 &= 0.5, && \text{length of 1st pendulum } (m) \\ l_2 &= 0.5, && \text{length of 2nd pendulum } (m) \\ g &= 9.82, && \text{acceleration of gravity } (m/s^2) \\ I_1 &= m_1 l_1^2/12, && \text{moment of inertia around pendulum midpoint} \\ I_2 &= m_2 l_2^2/12, && \text{moment of inertia around pendulum midpoint} \\ k &= 2.0, && \text{maximum control} \end{aligned}$$



(a) Lines show different independent trials that are halted once the system remains stable about the target state for one second.



(b) Diagram of the double pendulum system. Both joints are actuated.

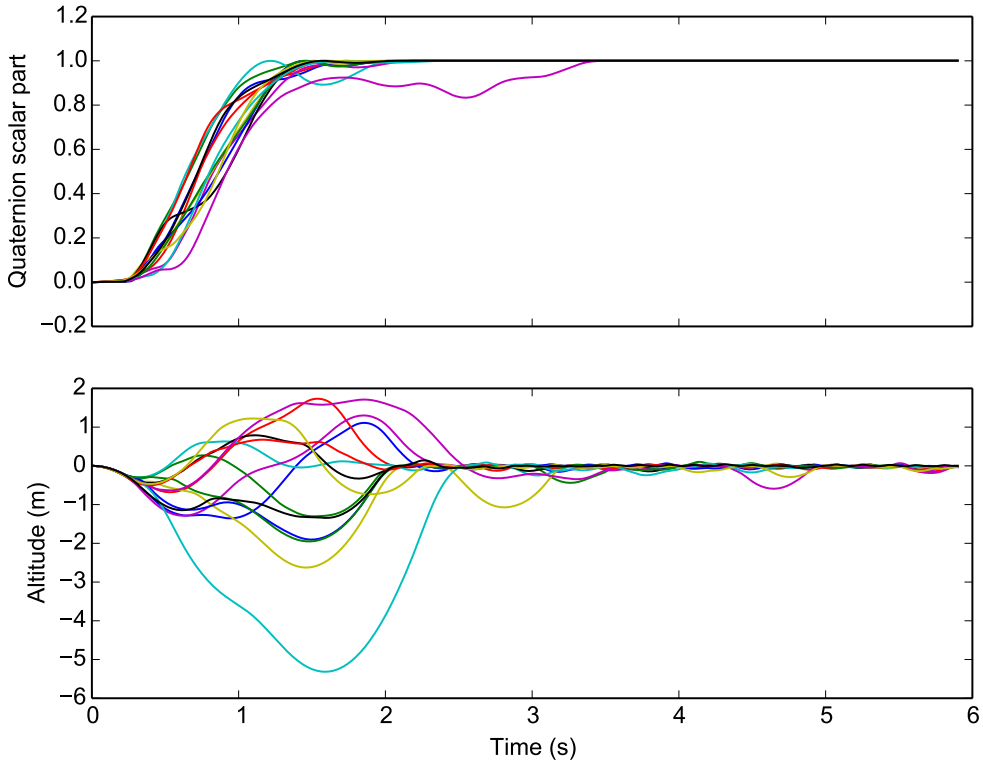
Figure 4.6: Double pendulum swing-up and balance experiments. The goal is to swing both pendulums up from the initial state hanging down to the upright balanced state. Torque constraints require building up energy by swinging.

The start state $[\dot{\theta}_1, \dot{\theta}_2, \theta_1, \theta_2]$ is sampled from a normal distribution with mean $[0, 0, \pi, \pi]$ and diagonal variance $[0, 0, .5, .5]$.

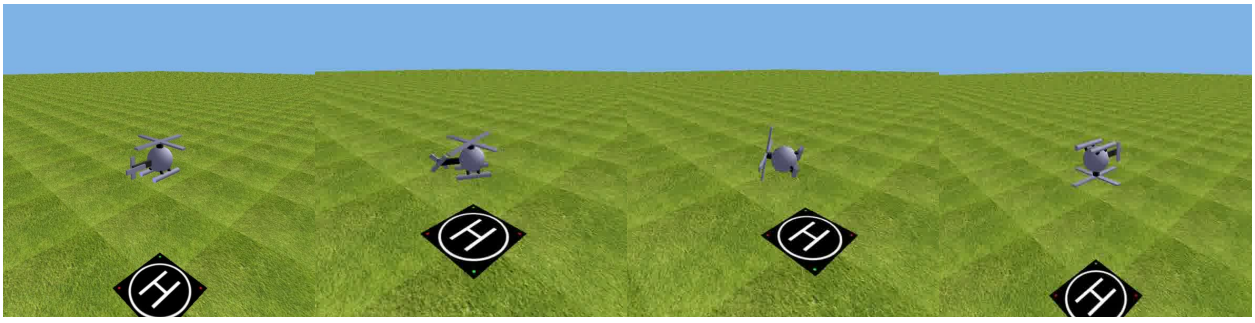
We modeled the system using a Dirichlet Process Mixture of Linear Models as described in Section 4.3. The model is set up to predict the angular accelerations both pendulums given sines and cosines of the pendulum angles, their angular velocities and the controls applied. Figure 4.6 shows a diagram of the system and a few different runs of our algorithm. It successfully learned the required model and solved the task after 18 ± 8 seconds of interaction. The competing Gaussian Process based approach [28] required a typical exploration time of $50s$ to achieve the same performance.

4.6.4 Helicopter Learning Inverted Hover.

We simulated a remotely controlled helicopter weighing about $5kg$, capable of producing at most $2.8g$ additional lift acceleration. The helicopter is initially in an upright position with zero velocity. The goal is to bring the system to rest at the same location but in *inverted* hover. Successfully completing the task requires learning to control the helicopter in a single flight, as no resets are available. Previous methods based on learning from demonstration[4] have solved this task before but, to the best of our knowledge, no other authors have had



(a) Lines show different independent trials. The visible jitter is the result of simulated gusts of wind that our planner learns to counteract correctly.



(b) Different stages of exploration progress after .2s, .6s, 1.1s and 2.1s respectively from left to right.

Figure 4.7: Helicopter learning inverted hover. The initial state (left) corresponds to upright hover but unknown dynamics. The goal state (right) is inverted hover.

a helicopter learn to perform such an acrobatic task without built-in prior knowledge or demonstrations.

The state of the helicopter is represented by four concatenated three dimensional vectors $[\dot{x}, w, x, r]$ where $\dot{x}$ and x are the velocity and the position in the inertial frame of reference, w is the helicopter's angular velocity in the helicopter's frame of reference and r represents the helicopter's orientation as unit rotation axis (Euler axis) times rotation angle. The corresponding equations of motion are:

$$
\begin{aligned}
0 = &- R^{-1}(r)\ddot{x} + \begin{bmatrix} b_4 & 0 & 0 \\ 0 & b_5 & 0 \\ 0 & 0 & b_6 \end{bmatrix} \cdot R^{-1}(r)\dot{x} + \begin{bmatrix} 0 \\ 0 \\ k_4 u_4 \end{bmatrix} + R^{-1}(r) \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \\
0 = &- \dot{w} + \begin{bmatrix} b_1 & 0 & 0 \\ 0 & b_2 & 0 \\ 0 & 0 & b_3 \end{bmatrix} \cdot w + \begin{bmatrix} k_1 u_1 \\ k_2 u_2 \\ k_3 u_3 \end{bmatrix} \\
0 = &- [R(r)w]_\times R^{-1}(r) + \sum_i \frac{\partial R^{-1}(r)}{\partial r_i} \cdot \dot{r}_i
\end{aligned}
$$

where $R(r)$ is rotation the matrix corresponding to state r. This matrix rotates the helicopter's frame of reference to the inertial frame of reference. The notation $[a]_\times$ represents the three dimensional cross product as a linear operator defined such that $[a]_\times \cdot b = a \times b$ for all $a, b \in R^3$. Parameters were chosen consistently with past work [4, 44], specifically:

$$
\begin{aligned}
k_1, k_2, k_3, k_4 &= 13.20, -9.21, 14.84, -27.5, && \text{control weights} \\
b_1, b_2, b_3 &= -3.47, -3.06, -2.58, && \text{translational friction parameters} \\
b_4, b_5, b_6 &= -.048, -.12, -.0005, && \text{rotational friction parameters} \\
g &= 9.81, && \text{acceleration of gravity } (m/s^2)
\end{aligned}
$$

In our chosen state space representation the helicopter's dynamics are approximately linear. Consequently we used the Bayesian linear model discussed in Section 4.2 to predict $R(r)^{-1}(\ddot{x} - [0, 0, g]^T)$ and $\dot{w}$ given $w, R^{-1}\dot{x}$ and controls u. The planning problem, however, is still non-linear and challenging since the state representation is not inertial. We assumed that rotational kinematics and the acceleration of gravity are known and do not need to be learned. Figure 4.7 shows a few examples of our algorithm successfully solving the task in 4.5 ± 1.5 seconds of interaction. The helicopter lost at most 6 meters of altitude in the learning process. For comparison, the best algorithm submitted to the 2008 Reinforcement Learning Competition for the helicopter hovering task [44] required 30 - 60 minutes of interaction.

4.7 Discussion

We extended Optimism Based Exploration, a tried-and-true reinforcement learning principle, to the context of continuous dynamical systems assumed to be locally linear. The approach is modular allowing use of any off-the-shelf optimal control planner for deterministic dynamics and any Bayesian dynamics model that provides prediction regions. In our experiments we used a Dirichlet Process Mixture Model whose complexity increases logarithmically in the size of the training data set.

Empirically we demonstrate data efficient learning on a number of tasks run in simulation: swing-up and balance for an under-actuated pendulum, cartpole and a double pendulum and learning inverted hover for a full state helicopter. Our method requires minimal tuning; the parameters were set to a-priori reasonable values and were not changed across different dynamical systems. We attribute the success of our method to a couple of factors:

- The exploration strategy employed is based on a principle that has theoretical support in various simpler reinforcement learning settings.
- The dynamics model we used was able to leverage the large volume of training data that is typically available in the context of our problem.

Chapter 5

Summary of Contributions

This dissertation focuses on three aspects of reinforcement learning that we believe to be important in practical applications: managing irreducible uncertainty by *risk aware planning*, ensuring *safe exploration* by restricting the set of allowable policies to avoid catastrophic failures, and reducing dynamics uncertainty by efficient *exploration* for continuous systems.

Risk aware planning is the typical approach in situations where dynamics uncertainty can not be reduced by further observations (that is when the true systems dynamics are stochastic). In Chapter 2 we explore this situation for discrete state space planning problems. The relevant key contributions are listed below:

- We propose a new objective for risk aware planning based on Chernoff bounds. Theorem 1 explains how this new objective relates to and subsumes a number of previously proposed risk aware planning objectives.
- Our risk-aware objective can be optimized efficiently by Algorithm 1, a newly proposed and theoretically justified method based on sequential exponential utility optimization.

Most exploration methods assume *ergodicity* when justifying performance; following any course of action the environment is assumed to allow return to the start state, so the possibility of catastrophic failure is "assumed away". In Chapter 3 we propose a method for enforcing ergodicity for the purpose of safe exploration in environments are not a priori ergodic. We do so by restricting the space of allowed policies so that ergodicity is provably preserved with high probability. An efficient algorithm for the case of discrete state space problems is presented. The relevant main contributions are listed below:

- We formulate a new definition of safety in relation to ergodicity, summarized by Equation 3.3. This definition allows enforcing safety for any reward-encoded objective including exploration.
- Algorithm 2 justified by Theorem 4 is a new method for enforcing safety per our definition for any reward based exploration method.

When dynamics uncertainty can be reduced by further observations it is important to take efficient exploratory actions. In Chapter 4 we investigate the possibility of efficient exploration for continuous dynamical systems, leading to a number of important contributions:

- We propose a new exploration algorithm based on the principle of *Optimism in the Face of Uncertainty* (Section 4.5) that is known to produce good results in simpler planning scenarios.
- Our experiments empirically validate the use of a classical Bayesian non-parameteric model, the *Dirichlet Process Mixture of Gaussians* (Section 4.3) for continuous dynamical systems in the context of exploration.

Bibliography

[1] MSL Landing Site Selection. User's Guide to Engineering Constraints, 2007.

[2] Stratigraphy of Potential Crater Hydrothermal System, 2008.

[3] Yasin Abbasi-Yadkori, Csaba Szepesvári, Sham Kakade, and Ulrike Von Luxburg. Regret Bounds for the Adaptive Control of Linear Quadratic Systems. In *Proceedings of the 24th Annual Conference on Learning Theory*, 2011.

[4] Pieter Abbeel, A. Coates, and Andrew Y. Ng. Autonomous Helicopter Aerobatics through Apprenticeship Learning. *The International Journal of Robotics Research*, June 2010.

[5] Rajeev Agrawal. Sample mean based index policies with O(log n) regret for the multi-armed bandit problem. In *Advances in Applied Probability*, volume 27, pages 1054 – 1078. Applied Probability Trust, 1995.

[6] Eitan Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

[7] Peter L. Bartlett Ambuj Tewari. Optimistic linear programming gives logarithmic regret for irreducible MDPs. In *Proceedings of Neural Information Processing Systems Conference*, 2007.

[8] Mauricio Araya, Olivier Buffet, and Vincent Thomas. Near-Optimal BRL using Optimistic Local Transitions. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 97–104, New York, NY, USA, July 2012. Omnipress.

[9] Anil Aswani and Patrick Bouffard. Extensions of Learning-Based Model Predictive Control for Real-Time Application to a Quadrotor Helicopter. In *Proc. American Control Conference (ACC)*, 2012.

[10] Anil Aswani, Patrick Bouffard, and Claire Tomlin. Extensions of Learning-Based Model Predictive Control for Real-Time Application to a Quadrotor Helicopter. In *ACC 2012, to appear*, Montreal, Canada, June 2012.

[11] Jean-Yves Audibert and Sébastien Bubeck. Regret Bounds and Minimax Policies under Partial Monitoring. *The Journal of Machine Learning Research*, 11:2785–2836, March 2010.

[12] Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, April 2009.

[13] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal on Computing*, 32(1):48–77, January 2002.

[14] David A. Benson, Geoffrey T. Huntington, Tom P. Thorvaldsen, and Anil V. Rao. Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method. *Journal of Guidance, Control, and Dynamics*, 29(6):1435–1440, November 2006.

[15] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, October 1996.

[16] John T Betts. *Practical methods for optimal control and estimation using nonlinear programming*, volume 19. Society for Industrial & Applied, 2010.

[17] David M. Blei and Michael I. Jordan. Variational Inference for Dirichlet Process Mixtures. *Bayesian Analysis*, 1(1):121–143, March 2006.

[18] Vincent D. Blondel and John N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, September 2000.

[19] VS Borkar and SP Meyn. Risk-sensitive optimal control for Markov decision processes with monotone cost. *Mathematics of Operations Research*, 27(1):192–209, 2002.

[20] M. Bouakiz and Y. Kebir. Target-level criterion in Markov decision processes. *Journal of Optimization Theory and Applications*, 86(1):1–15, July 1995.

[21] Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[22] Ronen I. Brafman and Moshe Tennenholtz. R-MAX - A General Polynomial Time Algorithm for Near-Optimal Reinforcement Learning. In *Journal of Machine Learning Research*, volume 3, pages 213–231, 2001.

[23] Eduardo F Camacho and Carlos Bordons. *Model predictive control*, volume 303. Springer Berlin, 1999.

[24] Sotirios P. Chatzis, Dimitrios Korkinof, and Yiannis Demiris. A nonparametric Bayesian approach toward robot learning by demonstration. *Robotics and Autonomous Systems*, 60(6):789–802, June 2012.

[25] Daniela Pucci de Farias and Benjamin Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51:2003, 2001.

[26] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: amazon's highly available key-value store. *ACM SIGOPS Operating Systems Review*, 41(6):205–220, 2007.

[27] B. Defourny, D. Ernst, and L. Wehenkel. Risk-aware decision making and dynamic programming. In *NIPS 2008 Workshop on Model Uncertainty and Risk in RL*, 2008.

[28] Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99(PrePrints), 2013.

[29] Marc Peter Deisenroth and Carl Edward Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning*, pages 465–472, Bellevue, Washington, USA, 2011. Omnipress.

[30] Marc Peter Deisenroth, Carl Edward Rasmussen, and Jan Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7):1508–1524, 2009.

[31] Erick Delage and Shie Mannor. Percentile optimization in uncertain Markov decision processes with application to efficient exploration. *ICML; Vol. 227*, page 225, 2007.

[32] Erick Delage and Shie Mannor. Percentile Optimization for Markov Decision Processes with Parameter Uncertainty. *Operations Research*, 58(1):203–213, 2010.

[33] Richard Durrett. *Probability: Theory and Examples*. Cambridge University Press, 2010.

[34] Aurélien Garivier and Olivier Cappé. The KL-UCB algorithm for bounded stochastic bandits and beyond. In *Proceedings of the 25th Annual Conference on Learning Theory*, 2011.

[35] A Geramifard, J Redding, N Roy, and J P How. UAV Cooperative Control with Stochastic Risk Models. In *Proceedings of the American Control Conference (ACC)*, San Francisco, CA, 2011.

[36] Jeremy H. Gillula and Claire J. Tomlin. Guaranteed safe online learning of a bounded system. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2979–2984. IEEE, September 2011.

[37] A Hans, D Schneegaß, AM Schäfer, and S Udluft. Safe exploration for reinforcement learning. In *ESANN 2008, 16th European Symposium on Artificial Neural Networks*, 2008.

[38] Matthias Heger. Consideration of risk in reinforcement learning. In *Proceedings of the 11th International Machine Learning Conference (1994)*, pages 105–111. Morgan Kaufmann, 1994.

[39] David H. Jacobson and David Q. Mayne. *Differential dynamic programming*. Modern analytic and computational methods in science and mathematics. American Elsevier Pub. Co., 1970.

[40] Philippe Jorion. *Value at risk: the new benchmark for managing financial risk*, volume 1. McGraw-Hill Professional, 2007.

[41] Michael Kearns and Satinder Singh. Near-Optimal Reinforcement Learning in Polynomial Time. *Machine Learning*, 49(2):209–232, November 2002.

[42] J. F. Kenney and E. S. Keeping. Cumulants and the cumulant-generating function, additive property of cumulants, and Sheppard's correction. In *Mathematics of Statistics*, chapter 4.10-4.12, pages 77–82. Van Nostrand, Princeton, NJ, 2 edition, 1951.

[43] J. Zico Kolter and Andrew Y. Ng. Near-Bayesian exploration in polynomial time. In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, pages 1–8, New York, New York, USA, 2009. ACM Press.

[44] Rogier Koppejan and Shimon Whiteson. Neuroevolutionary reinforcement learning for generalized helicopter control. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, GECCO '09, pages 145–152, New York, NY, USA, 2009. ACM.

[45] Yann Le Tallec. *Robust, Risk-Sensitive, and Data-driven Control of Markov Decision Processes*. PhD thesis, Massachusetts Institute of Technology, 2007.

[46] Steve Levitt and Adi Ben-Israel. On Modeling Risk in Markov Decision Processes. *Optimization and Related Topics*, pages 27–41, 2001.

[47] Lihong Li, Michael L. Littman, and Thomas J. Walsh. Knows what it knows: a framework for self-aware learning. In *Proceedings of the 25th international conference on Machine learning*, pages 568–575, 2008.

[48] Weiwei Li and Emanuel Todorov. Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems. In *First International Conference on Informatics in Control, Automation and Robotics*, pages 222–229, 2004.

[49] Mary Kae Lockwood. Introduction: Mars Science Laboratory: The Next Generation of Mars Landers. *Journal of Spacecraft and Rockets*, 43(2), 2006.

[50] Shie Mannor and John N. Tsitsiklis. Mean-Variance Optimization in Markov Decision Processes. In *Proceedings of the 28 International Conference on Machine Learning*, 2011.

[51] S.I. Marcus, E. Fernández-Gaucherand, D. Hernández-Hernandez, S. Coraluppi, and P. Fard. Risk sensitive Markov decision processes. *Systems and Control in the Twenty-First Century*, 29:263–281, 1997.

[52] Teodor Mihai Moldovan and Pieter Abbeel. Risk Aversion in Markov Decision Processes via Near-Optimal Chernoff Bounds. In P. Bartlett Weinberger, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q., editors, *Advances in Neural Information Processing Systems 25*, pages 3140 – 3148, Lake Tahoe, NV, USA, 2012.

[53] Teodor Mihai Moldovan and Pieter Abbeel. Safe Exploration in Markov Decision Processes. In *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, UK, 2012.

[54] Manfred Morari and Jay H Lee. Model predictive control: past, present and future. *Computers & Chemical Engineering*, 23(4):667–682, 1999.

[55] Arnab Nilim and Laurent El Ghaoui. Robust Control of Markov Decision Processes with Uncertain Transition Matrices. *Operations Research*, 53(5):780–798, 2005.

[56] F Palacios-Gomez, L Lasdon, and M Engquist. Nonlinear Optimization by Successive Linear Programming. *Management Science*, 28(10):1106–1120, 1982.

[57] Jason Pazis and Ronald Parr. Non-Parametric Approximate Linear Programming for MDPs. In Wolfram Burgard and Dan Roth, editors, *Twenty-Fifth Conference on Artificial Intelligence (AAAI-11)*. AAAI Press, 2011.

[58] I. Michael Ross and Mark Karpenko. A review of pseudospectral optimal control: From theory to flight. *Annual Reviews in Control*, 36(2):182–197, December 2012.

[59] Jay K. Satia and Roy E. Lave Jr. Markovian Decision Processes with Uncertain Transition Probabilities. *Operations Research*, 21(3):728–740, 1973.

[60] Alexander L. Strehl and Michael L. Littman. A theoretical analysis of Model-Based Interval Estimation. In *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pages 856–863, New York, New York, USA, August 2005. ACM Press.

[61] Tilo Strutz. *Data Fitting and Uncertainty: A practical introduction to weighted least squares and beyond*. Vieweg+Teubner Verlag, 2010.

[62] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction*. MIT Press, 1998.

[63] Yee Whye Teh. Dirichlet Process. In Claude Sammut and Geoffrey I Webb, editors, *Encyclopedia of Machine Learning*, pages 280–287. Springer, 2010.

[64] Oskar Von Stryk. *Numerical solution of optimal control problems by direct collocation*. Citeseer, 1991.

[65] Congbin Wu and Yuanlie Lin. Minimizing Risk Models in Markov Decision Processes with Policies Depending on Target Values. *Journal of Mathematical Analysis and Applications*, 231(1):47–67, 1999.

Appendix A

Additional Proofs and Examples

A.1 Example MDP where optimizing cost percentiles is counter-intuitive

We thank the anonymous reviewer that brought to our attention a simple MDP where optimizing percentiles leads to some counter intuitive choices. The MDP is represented in detail in Figure A.1. There only one choice of actions, in state S_3, which is transient, and does not lead to state S_1. However, the example shows that the optimal policy, with respect to $F_{0.1}^{-1}[V(\pi)]$, depends on the parameters of state S_1. This is counter-intuitive because, once we are in state S_3, the decision of which action to take should not depend on the parameters of state S_2, which we know we will never reach, since we are now in S_3.

A.2 Proof of Theorem 1

We will use the following properties of *cumulant generating function*, $f(z) := \log Ee^{zJ}$, [42]:

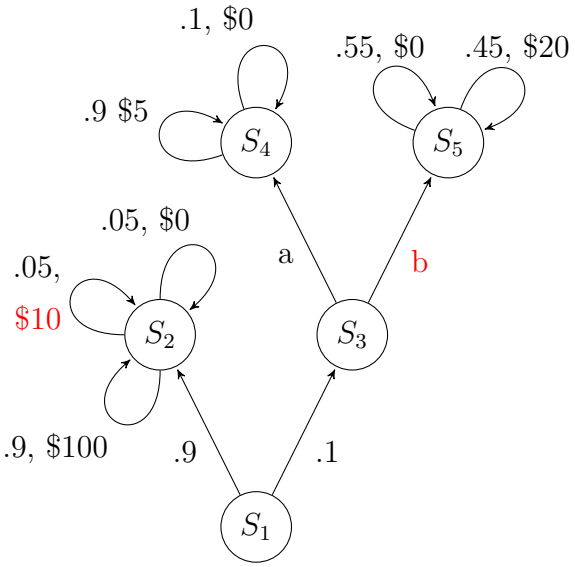
a) $f(z) = \sum_{i=1}^{\infty} z^i k_i/i!$ where k_i are the *cumulants* of J. Particularly, $k_1 = E[J]$, $k_2 = \text{Var}[J]$.

b) $f(z)$ is a strictly convex function, analytic and infinitely differentiable in a neighborhood of zero, if it is finite in that neighborhood.

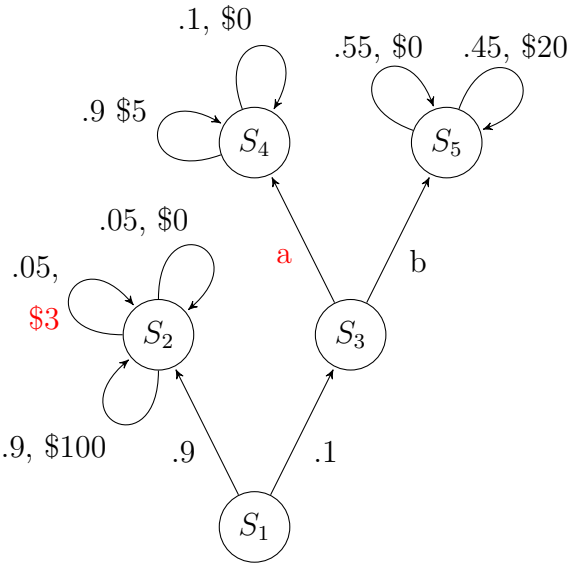
Our first goal is to show that the Chernoff functional is well defined. Note that we can write the definition in the equivalent form:

$$C^{\delta}[J] = \inf_{\theta>0}(\theta f(\theta^{-1}) - \theta \log \delta) = \inf_{z>0} z^{-1}(f(z) - \log \delta) \tag{A.1}$$

where f is a strictly convex function by property b. Note that the first term, $\theta f(\theta^{-1})$, is the perspective transformation of a strictly convex function, and, thus, is also strictly convex



(a) The optimal policy is taking action b, for which $F_{0.1}^{-1}[V(\pi)] = 10$.



(b) Now the optimal policy is taking action a, for which $F_{0.1}^{-1}[V(\pi)] = 5$

Figure A.1: Example showing that percentile optimization for MDPs is sometimes counter-intuitive. We are optimizing $F_{0.1}^{-1}[V(\pi)]$. Changing parameters of state S_2 affects the policy in state S_3 even though S_2 is unreachable from S_3.

[21], while the second term is linear, so also convex. This implies that the optimization problem in θ is strictly convex, and, consequently, the optimum is well defined.

We show property i by using Markov's inequality on the positive random variable $e^{J/\theta}$:

$$P(J \geq j) = P\{e^{J/\theta} \geq e^{j/\theta}\} \leq e^{-j/\theta} E e^{J/\theta} \quad \forall j, \theta > 0$$

Solve for j after setting the right hand side to δ:

$$j = \theta(\log E e^{J/\theta} - \log \delta) \quad \text{and} \quad P(J \geq \theta(\log E e^{J/\theta} - \log \delta)) \leq \delta$$

Since this is true for all $\theta > 0$, it will also be true for $\theta^* = \operatorname{argmin}_{\theta>0} \theta(\log E e^{J/\theta} - \log \delta)$, which shows that $P(J \geq C^\delta[J]) \leq \delta$.

To show property ii, first note that $C^1[J] = \inf_{\theta>0} \theta f(\theta^{-1})$. Using the Taylor series expansion of f, property a:

$$\frac{d}{d\theta}(\theta f(\theta^{-1})) = f(\theta^{-1}) - \frac{f'(\theta^{-1})}{\theta} = \sum_{i=1}^{\infty} (\theta^{-i} + i\theta^{-i-2}) \frac{k_i}{i!} \quad \Rightarrow \quad \lim_{\theta\to\infty} \frac{d}{d\theta}(\theta f(\theta^{-1})) = 0$$

Since $\theta f(\theta^{-1})$ is strictly convex, we conclude that $\inf_{\theta>0} \theta f(\theta^{-1})$ must occur at $\theta^* = \infty$. Using the Taylor series expansion again, we can see that $\lim_{\theta\to\infty} \theta f(\theta^{-1}) = k_1 = E[J]$, and, consequently, $C^1[J] = \inf_{\theta>0} \theta f(\theta^{-1}) = \lim_{\theta\to\infty} \theta f(\theta^{-1}) = E[J]$.

To show property iii, let $F_\delta^{-1}[J] = \sup\{j : P\{J \geq j\} > \delta\}$. For now assume this is finite for all $\delta \in [0, 1)$. Let $Y = J - F_0^{-1}[J]$, so $P\{Y \leq 0\} = 1$ and $e^{Y/\theta} \leq 1$ almost surely. Then $C^\delta[J] = C^\delta[Y] + F_0^{-1}[J]$. The left hand side in the equation below is a restatement of property i:

$$\begin{aligned} F_\delta^{-1}[Y] \leq C^\delta[Y] = \inf_{\theta>0} \theta(\log E e^{Y/\theta} - \log \delta) \leq \lim_{\theta\to 0} \theta(\log E e^{Y/\theta} - \log \delta) = \\ = \lim_{\theta\to 0} \theta \log E e^{Y/\theta} \leq 0 \quad \text{since} \quad E e^{Y/\theta} \leq 1 \end{aligned}$$

In this equation, take the limit $\delta \to 0$ and we get that $0 = F_0^{-1}[Y] \leq C^0[Y] \leq 0$, so we can conclude $C^0[J] = F_0^{-1}[J]$. If $F_0^{-1}[J] = \infty$, then, by the fact that $F_0^{-1}[Y] \leq C^0[Y]$ we get $C^0[Y] = \infty$.

Property v. As we saw in the proof of property ii, as $\delta \to 1$, $\theta^* \to \infty$, so we can approximate the cumulant generating function by its first two Taylor series terms: $f(\theta^{-1}) = \log E e^{J/\theta} \approx E[J]/\theta + \operatorname{Var}[J]/(2\theta^2)$. Using this form we can solve the optimization problem exactly, and we get $C^\delta[J] \approx E[J] + \sqrt{2\log(1/\delta)\operatorname{Var}[J]}$. Property iv follows immediately, as all the cumulants of the Gaussian distribution, except the first two, are zero [42].

We will show the first part of property vi, smoothness, by using the implicit function theorem. Let $g(\delta) = \operatorname{argmin}_{z>0} z^{-1}(f(z) - \log \delta)$. As we have seen, this function is well defined, so it is the only value of z where the derivative of the expression to zero:

$$\frac{d}{dz}(z^{-1} f(z) - z^{-1} \log \delta) = z^{-2}(z f'(z) - f(z) + \log \delta) = z^{-2} h(\delta, z).$$

In other words, $g(\delta)$ is also implicitly defined by the equation $h(\delta, g(\delta)) = 0$. The function h is smooth since the cumulant generating function, f, is smooth, so, by the implicit function theorem, $g(\delta)$ is also smooth. To show monotonicity, let $r_z(\delta) = z^{-1}(f(z) - \log \delta)$. This a class of decreasing functions of δ, indexed by z. The point-wise infimum over a class of decreasing functions is also decreasing, so $C^\delta[X] = \inf_{z>0} r_z(\delta)$ is decreasing in δ.

A.3 Proof of Theorem 2

We start by restating the previous definition of $n(\varepsilon)$ from Section 2.4.3:

$$n(\varepsilon) = \left\lceil \frac{(j_M - j_m)}{\varepsilon} \log\left(\frac{\theta_2}{\theta_1}\right) + \log\left(\frac{\theta_2}{\theta_1}\right) \right\rceil.$$

By definition of the ceiling function and the basic inequality $\log(1 + x) > x/(x + 1)$, we get that:

$$n(\varepsilon) \geq \frac{\varepsilon + (j_M - j_m)}{\varepsilon} \log\left(\frac{\theta_2}{\theta_1}\right)$$

$$\Rightarrow \quad \log\left(\frac{\varepsilon}{j_M - j_m} + 1\right) > \frac{\varepsilon/(j_M - j_m)}{\varepsilon/(j_M - j_m) + 1} = \frac{\varepsilon}{\varepsilon + (j_M - j_m)} \geq \frac{1}{n(\varepsilon)} \log\left(\frac{\theta_2}{\theta_1}\right).$$

Exponentiating both sides we get:

$$\frac{\varepsilon}{j_M - j_m} + 1 > \left(\frac{\theta_2}{\theta_1}\right)^{1/n} \quad \Leftrightarrow \quad \varepsilon > (j_M - j_m)\left(\left(\frac{\theta_2}{\theta_1}\right)^{1/n} - 1\right).$$

This inequality and Equation (2.4) imply that:

$$0 \leq f(\theta_i^{n(\varepsilon)}) - f(\theta_{i+1}^{n(\varepsilon)}) < \varepsilon.$$

A.4 Proof of Theorem 3.

Proof. We will prove the theorem by reducing the satisfiability problem in conjunctive normal form with three variables (3SAT) to the problem of deciding whether there exists a p-safe policy for a belief that we will construct. The 3SAT problem amounts to deciding whether there exists an assignment to boolean variables $\{U_k\}$ such that the following expression is true:

$$(X_1 \vee Y_1 \vee Z_1) \wedge \cdots \wedge (X_n \vee Y_n \vee Z_n)$$

where each of the variables X_i, Y_i, Z_i equals one of the variables in $\{U_k\}$, possibly negated.

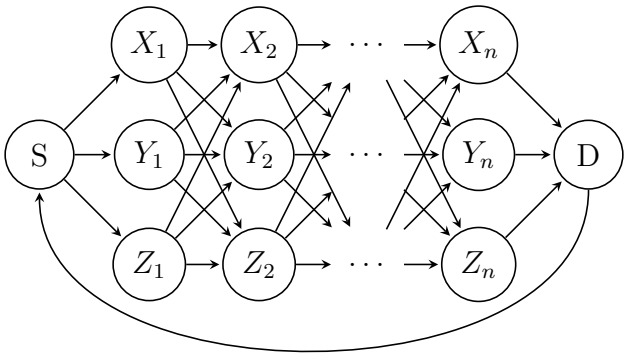


Figure A.2: MDP reduction of the 3SAT problem.

We start by constructing an MDP to represent this problem as shown in Figure A.2. In addition to actions corresponding to the outgoing arrows, the agent also has the option of remaining in the same state. A transition from some state to another state will succeed if and only if the boolean variable corresponding to the origin state is true. The boolean variable associated to states S and D are always true. Our belief is the uniform distribution over truth values of the boolean variables $\{U_k\}$.

Now consider the following simple policy: from D go to S and then stay in S. For any recall time $T > 0$, the recall event will find the agent in state S, so the policy is p-safe for any $p > 0$ if and only if the belief assigns a non-zero measure to MDPs in which state D is accessible form state S, so if any only if there exists at least one boolean assignment for the $\{U_k\}$ such that state D is accessible from S. It is easy to see that, this is the case if and only if the 3SAT formula is satisfied, and this observation completes the reduction.

This result should come as no surprise since similar optimization problems have been shown to be NP-hard in the context of *Partially Observable Markov Decision Processes* [18]. □

A.5 Proof of Theorem 4.

Proof. The result is an immediate consequence of the following Lemma. □

Lemma 5. *Given a belief β and a policy π, there exists a policy dependent reward correction, $\sigma^{\beta,\pi}$, defined below, such that the MDP with transition measure $p := E_\beta P$ and rewards $r + \sigma^{\beta,\pi}$, where $r := E_\beta R$, has the same expected total return as the belief for any initial*

distribution. Formally:

$$\forall \rho \quad E_\beta E^P_{\rho,\pi} \sum_{t=0}^{\infty} R_{S_t,A_t} = E^p_{\rho,\pi} \sum_{t=0}^{\infty} \left(r_{s,a} + \sigma^{\beta,\pi}_{s,a}\right)$$
$$\sigma^{\beta,\pi}_{s,a} := \sum_{s'} E_\beta \left[(P_{s,a,s'} - E_\beta[P_{s,a,s'}]) E_{s',\pi,P}[V]\right].$$

Proof. The Markov property under belief β reads:

$$E_\beta E^P_{s,\pi}[V] = \sum_a \pi_{s,a} E_\beta[R_{s,a}] + \sum_a \pi_{s,a} \sum_{s'} E_\beta[P_{s,a,s'} E^P_{s',\pi}[V]].$$

The Markov property assuming expected transition frequencies and expected rewards with safety penalty is:

$$E^p_{s,\pi}[\bar{V}] = \sum_a \pi_{s,a}(r_{s,a} + \sigma^{\beta,\pi}_{s,a}) + \sum_a \pi_{s,a} \sum_{s'} p_{s,a,s'} E^p_{s',\pi}[\bar{V}]].$$

Now let $\Delta_s := E_\beta E^P_{s,\pi}[V] - E^p_{s,\pi}[\bar{V}]$. By subtracting the first two equations we get that:

$$\Delta_s = \sum_a \pi_{s,a} \sum_{s'} p_{s,a,s'} \Delta_{s'}.$$

We can see that Δ satisfies the same equation as the value function in an MDP with transition measure p and zero rewards. Since the value function in such an MDP is uniquely defined and identically zero, we conclude $\Delta_s = 0$. □