Efficient Stochastic Control with Kullback Leibler Costs using Kernel Methods

Sacha Sokoloski
Bernstein Center for Computational Neuroscience Berlin

Abstract

Contemporary research in optimal control has yielded many elegant solutions to problems of manipulation and locomotion. Nevertheless, in the case of stochastic control, basic problems continue to resist general solutions.

This thesis is focused upon a restricted set of stochastic control problems which are nevertheless sufficient for modelling general problems in robot kinematics. The restrictions on the stochastic equations of motion simplify accompanying control equations, allowing for more efficient numerical solutions.

In order to model the system dynamics, we rely on recent work in the application of kernel methods to modelling conditional expectations, and the application of these methods to evaluating solutions of fundamental control equations. Combining this work with the restricted stochastic equations yields efficient solutions to basic problems of robot kinematics.

Neueste Forschungsergebnisse aus dem Bereich der optimalen Steuerung lieferten viele elegente Lösungen für Manipulations- und Lokomotionsprobleme. Im Fall stochastischer Steuerung fehlen jedoch weiterhin allgemeine Lösungen für grundlegende Probleme.

Diese Arbeit befasst sich mit der eingeschränkten Steuerung einer Klasse von allgemeinen stochastischen Problemen der Roboterkinematik. Die Beschränkungen der stochastischen Bewegungsgleichungen vereinfachen die Steuerungsgleichungen und ermöglichen effiziente numerische Lösungen.

Wir verwenden Kernel-Methoden um die bedingten Erwartungswerte der Systemdynamik zu modellieren sowie zur Lösung der Steuerungsgleichungen. Die Kombination dieser Methoden mit den eingeschränkten stochastischen Gleichungen liefert effiziente Lösungen für grundlegende Probleme der Roboterkinematik.

Contents

1 Introduction

There are many formalisms upon which one may draw in order to model and solve control problems, each of which comes with its own advantages and disadvantages. On one hand, if one simply wishes for particular optimal solutions to well defined systems, as will usually be the case in e.g. transportation and industrial robotics, then the flexibility and adaptability of the agent will be more or less irrelevant, and the environment will usually be tightly controlled in order to further simplify the problem. On the other hand, if one wishes to control systems in the way that animals control their own bodies, then a different set of tools is required.

If we consider some of our most successful achievements in engineered locomotion, e.g. cars and planes, we achieve speed and efficiency by keeping the underlying mechanisms simple, and complementing them with an environment in which they ideally operate. In the case of cars, although wheels are very limited in the kinds of terrain which they can successfully navigate, by building roads we can provide ideal conditions for wheel based locomotion where needed.

In the case of planes, although aerodynamics in general is highly nonlinear, we sidestep this problem by building planes with purpose built shapes, and controlling the basic tasks of take off, flight, and landing such that the system dynamics remains within linear regimes. In this way we can reliably fly across the Atlantic ocean in a matter of hours. Nevertheless, when it comes to the tasks of turning quickly or landing in highly constrained environments, our machines pale in comparison to the elegance of birds and bats.

Up until recently, most of our attempts at animal/human like locomotion have resulted in machines which are both inflexible and inefficient. This is largely because we have tried to solve the control problem in the same way as we have with cars and planes: by carefully structuring the environment, and keeping the dynamics within linear regimes. The well known robot Asimo, for all the ways in which it is an impressive achievement of engineering, cannot successfully operate outside of a few well designed scenarios. Moreover, Asimo consumes energy at roughly 20 times the rate of a human in order to maintain a walking gait [11].

The rigid body dynamics of animal bodies are generally well understood, and our ability to build complex robot bodies is rapidly expanding. Where the primary cause of our inability to control nonlinear systems arises is in the limits of modern control theory itself. In this thesis, we will combine a number of

recent developments in machine learning and control theory in order to solve certain fundamental nonlinear control problems in what we believe to be a biologically plausible, 'realistic' way.

By realistic we mean essentially three things. Firstly, that realistic systems operate in the presence of noise. Even though the magnitude of this noise may be quite small, its presence is nevertheless important. In general, the presence of noise necessitates feedback based control, thus barring highly tuned, open loop solutions.

Secondly, that realistic control, at least in the case of low level robotics, is best formulated as an infinite time horizon problem rather than a finite time horizon one. Generally speaking, finite time horizon solutions rely on knowing the cost of states at a final time, from which we can recursively compute the cost of available decisions at the current time, which in turn allows us to behave optimally. Although such a framework is certainly appropriate for high level planning tasks, we feel that for basic problems of gesture and locomotion, control should be driven purely by contemporary feedback rather than by developing, in advance, the complete plan for the desired action.

Finally, in order to estimate the cost of its actions, an agent requires an understanding of the dynamics of its body and environment. Although we are indeed simulating these systems, and could provide the equations of motion directly to the agent, we doubt that animals generally have fully tuned equations of motion built into their nervous systems. Rather, realistic systems sample their environment and build models from their experience.

The subsequent sections of this thesis are organized as follows: In chapter 2 we review basic rigid body kinematics, and reinterpret equations of motion as stochastic differential equations. In chapter 3, we review optimal control and the Bellman equations, and introduce the so called KL control framework and its linearized form. In chapter 4 we review recent work in estimating expected values by embedding probability distributions in reproducing kernel Hilbert spaces, and show how these embeddings can be used to model the dynamics of our stochastic systems. In chapter 5 we apply our ability to model system dynamics toward developing a series of efficient algorithms for estimating solutions to the Bellman equations. In chapter 6 we present the results of our control simulations, and in chapter 7 we will consider ways in which this work can be improved, as well as place this work within a larger scientific context.

2 Stochastic Robotics

If we focus upon earth bound locomotion it is easy to see that animals bodies, being composed of hundreds of joints and actuators, are both far more complex than present day machines, and far more flexible. What is remarkable is that this complexity does not prevent animals from also achieving high degrees of efficiency. In order to achieve the successes of animals, we must confront rather than hide from the nonlinear dynamics of complex mechanical systems. Thankfully, the dynamics of such systems are well understood.

The dynamics of animal bodies can be modelled as a series of joints and bones, and as such the dynamics of realistic robot control falls almost entirely within the field of rigid body dynamics. Moreover, many motions such as reaching and walking can be understood as achieving and stabilizing fixed points and limit cycles, and dynamical systems theory thus provides the tools we need in order to formally characterize the goals of our actions. We provide a brief review of these topics in section 2.1.

Another point which is critical, but less well understood, is the role of noise in dynamics and control. Intuitively it is clear that we cannot execute actions with perfect precision, and as such even routine motions such as walking will make us quite nervous, if where we are walking is up a mountain along a nar-

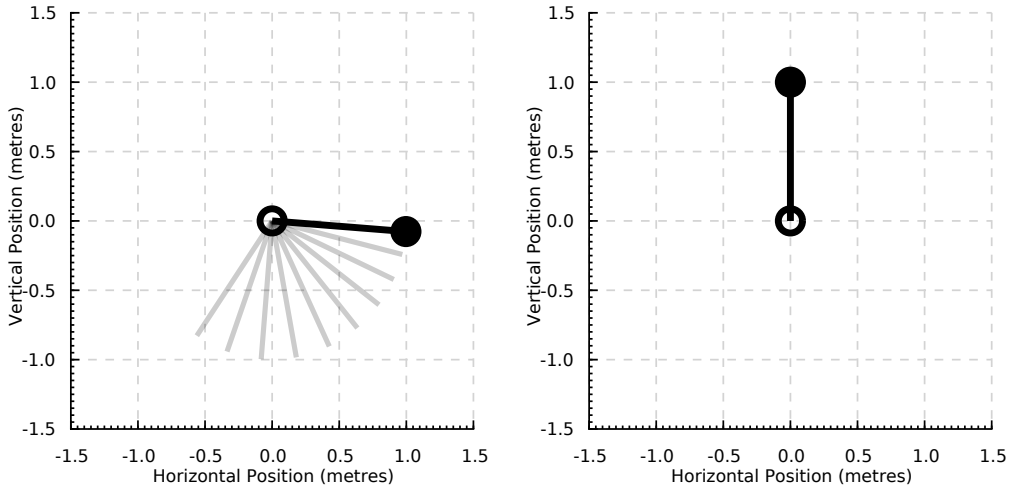

Figure 2.1: **The Simple Pendulum**: Left) A pendulum in motion. Right) A pendulum at rest at an unstable fixed point.

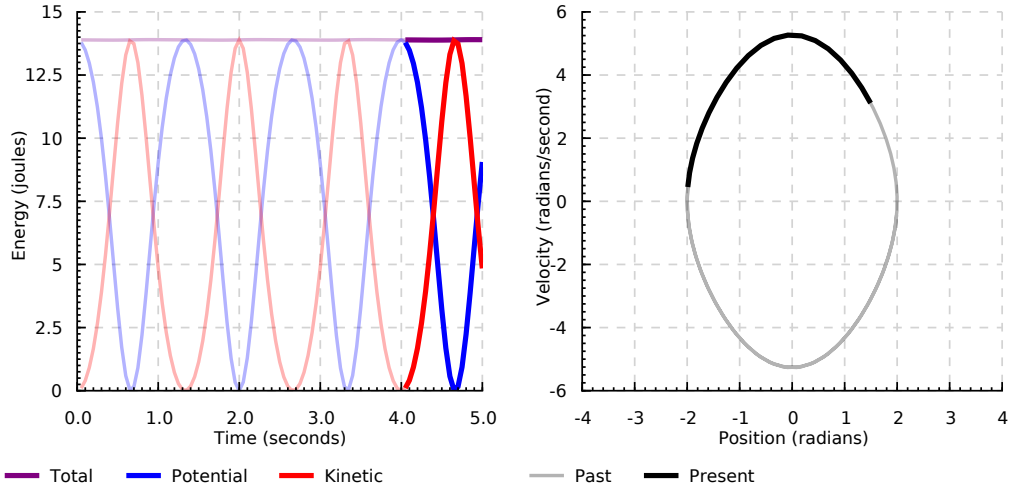

Figure 2.2: **Conservative Dynamics**: In this simulation, $l = 1m, m = 1kg$, and $g = 9.81N(m/kg)^2$. Left) A plot of the energies over time. In a conservative system, the total energy of the system remains constant as the kinetic and potential energies are exchanged. Right) The phase portrait of the pendulum. In a conservative system the phase portrait traces a circle.

row cliff. In section 2.2 we review basic stochastic differential equations, and then combine it with our rigid body dynamics in order to offer a preliminary model of robot dynamics in the presence of noise.

2.1 Robot Kinematics

Consider the simple pendulum depicted in figure 2.1, defined by the following equation of motion (2.1):

$$ml^2\ddot{\theta} + mgl\sin\theta = Q, \tag{2.1}$$

where m is the mass and l the length of the pendulum, θ is the angle of the pendulum measured counter-clockwise from its lowest point, g is the gravitational constant, and Q represents the generalized forces acting upon the system. When $Q = 0$, no forces are acting on the system, energy is conserved, and the pendulum will repeat its perfect trajectory in perpetuity (figure 2.2).

Sadly, our world is not perfect, and Q will only be 0 when the various forces of which it is composed are in balance. In this deterministic setting, we are interested in two forces, namely damping and control. Therefore in general we write

$$Q = u - b\dot{\theta}, \tag{2.2}$$

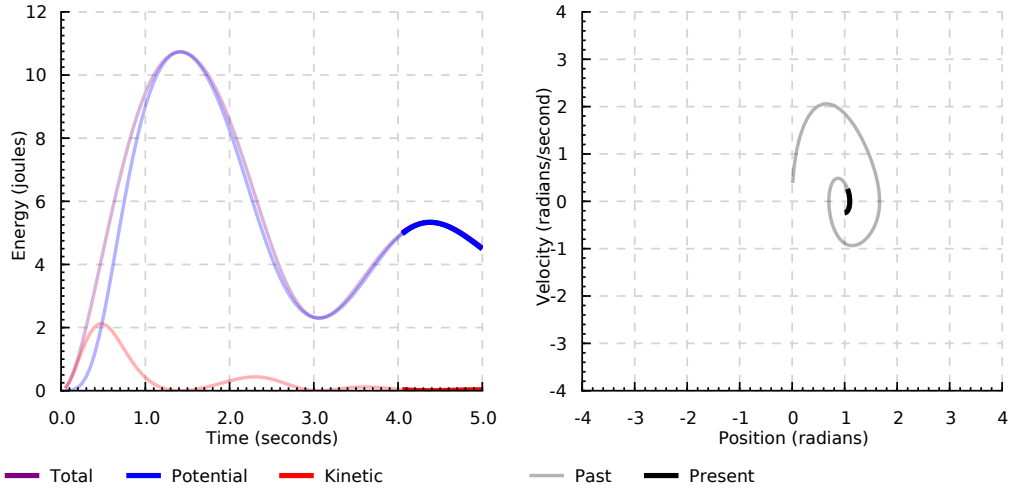

Figure 2.3: **Reaching a Fixed Point**: In this simulation, $u = 8J$ and $b = 1 \cdot m \cdot kg/s$. Left) The generalized forces Q change the total energy of the system. Control drives the system to a particular potential energy, and damping drives the kinetic energy to 0. Right) The system is tending towards a stable fixed point at $\theta \approx 1$.

where u is the control, and $b > 0$ is the damping coefficient.

In this context, the pendulum is said to be in *mechanical equilibrium* when $\ddot{\theta} = \dot{\theta} = 0$. Newton's first law of motion implies that if the pendulum is in mechanical equilibrium, it will remain there until acted on by an external force.

The locations where the system is in mechanical equilibrium are referred to as *fixed points*. When $u = 0$, the fixed points of the pendulum are at $\theta = 0$ and $\theta = \pi$. These two fixed points differ in an important way however, namely that the fixed point at 0 is *stable*, and the one at π *unstable* (see figure 2.1, Right). Formally, we define a fixed point x^* as *Lyapunov stable* if and only if

$$\forall \epsilon, \exists \delta, \text{ s.t. } |x(0) - x^*| < \delta \Rightarrow |x(t) - x^*| < \epsilon, \forall t \tag{2.3}$$

Intuitively this means that around x^* there is a *basin of attraction* such that if the system is within the basin, it will approach the fixed point over time. Unstable fixed points however have no basin of attraction, and if a system at an unstable fixed point is perturbed, it will not naturally return to the fixed point.

When $u \neq 0$, this has the effect of moving and/or removing the system's fixed points. Therefore, if we wish for the pendulum to 'reach' for a certain point in space, the right way characterize this goal is as stabilizing a fixed point around the point for which we wish to reach, by counter-balancing the force due to gravity with the controlled force (figure 2.3).

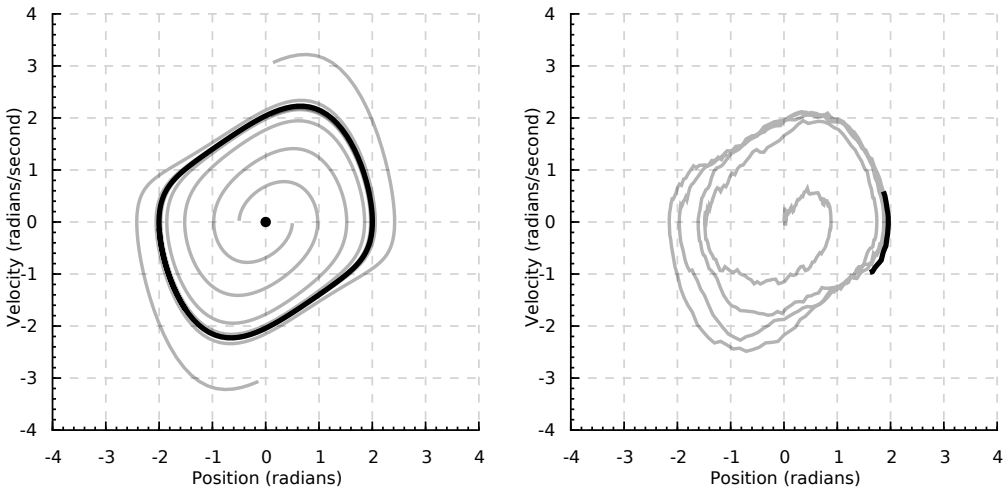

Figure 2.4: **The Van der Pol Oscillator**: Left) Phase portrait of the Van der Pol oscillator with $b = 0.5 \cdot m \cdot kg/s$. The black circle indicates the limit cycle and the dark point indicates the unstable fixed point. The softer lines are sample trajectories. Right) A stochastic Van der Pol oscillator with $b = 0.5 \cdot m \cdot kg/s$ and $\sigma^0 = 0.2J$, and initial conditions $q = \dot{q} = 0$. The trajectory is perturbed from the fixed point, and then evolves stochastically along the limit cycle.

Another highly relevant concept from dynamical systems is that of the *limit cycle*. Limit cycles are more difficult to formally characterize than fixed points, but intuitively they are clear. Consider one of the simplest systems which exhibits a stable limit cycle, namely, the *Van der Pol oscillator* with equation of motion

$$\ddot{q} + b(q^2 - 1)\dot{q} + q = 0, \tag{2.4}$$

where b can be interpreted as a nonlinear damping coefficient. We provide simulations of this system in the left panel of figure 2.4, where what is formally difficult to characterize becomes readily apparent: (almost) regardless of the initial conditions, as time progresses the trajectories of the system converge. In general, if systems within a certain basin tend toward a trajectory rather than simply a fixed point, this trajectory is referred to as a limit cycle.

Limit cycles are fundamental to understanding walking and other forms of locomotion. Locomotions in general are high dimensional periodic orbits in joint space, and particular gaits can be modelled as high dimensional limit cycles. If a robot can stabilize the appropriate limit cycle, it can walk. Indeed, as is suggested in the right panel of figure 2.4, it can walk even in the presence of noise.

2.2 Stochastic Dynamics

In order to add stochasticity to our equations of motion, we draw upon the mathematics of stochastic differential equations (SDEs). The solution of an SDE is a stochastic process, but stochastic processes are not generally differentiable. Stochastic processes may, however, be continuous, and therefore we may describe them via integral equations. For our purposes, the stochastic integral equation of interest is that of the *controlled Ito diffusion*:

$$x(t+dt) - x(t) = \int_t^{t+dt} \mu(x(s), u(s))ds + \int_t^{t+dt} \sigma(x(s), u(s))d\omega(s), \quad (2.5)$$

where t is a time and dt a time increment, μ and σ are functions known as the drift and diffusion, respectively, ω is a white noise process, which is the stochastic derivative of a Wiener process $d\omega$, u is the control signal, and x, the solution, is some stochastic process. The core property of the Wiener process is that its increments $d\omega(t+dt) - d\omega(t)$ are normally distributed random variables with mean 0 and variance t.

Equation 2.5, though formally sound, is also rather cumbersome. SDEs are typically written more concisely in the following form:

$$dx = \mu dt + \sigma d\omega. \quad (2.6)$$

Since this is a first order equation, before we can add noise to our pendulum we must decouple equation 2.1

$$\begin{aligned} \frac{d\theta}{dt} &= \dot{\theta}, \\ \frac{d\dot{\theta}}{dt} &= \frac{Q - mgl\sin\theta}{ml^2}. \end{aligned} \quad (2.7)$$

Ready to add stochasticity to our equations of motion, we must still consider where amongst these equations we ought to do this, and the nature of the noise which we are attempting to model. Since we do not wish to fundamentally change the nature of our underlying equations of motion, it would be most prudent to model noise as an external force and therefore add a noise term to Q.

Now conceptually, we may separate sources of noise into environmental sources and control sources. On one hand, control noise results from a variety of factors, for example the stochastic nature of control signals in the nervous system, and inaccuracies in transforming these signals into muscle responses.

Since there are a large number of indiscriminate factors contributing to this noise, modelling it as white noise is appropriate.

On the other hand, sources of environmental noise are more difficult to quantify. An obvious candidate for a source of environmental noise would be strong winds, but external perturbations due to wind are better understand as chaotic rather than stochastic - patterns of wind have a great deal of structure, despite being unpredictable. In order to approximate environmental perturbations via stochastic processes we would likely need a more complicated process than white noise, which would significantly complicate our mathematics.

Therefore, in this preliminary treatment of stochastic robot kinematics, our primary application of stochasticity is in modelling the imprecision of control. Nevertheless, we leave open the possibility that if a robot is attempting to act in a highly disruptive environment, then as a first approximation it can model these effects as a degradation of its control. If we imagine the cautious motions of a person attempting to cross a rickety bridge on a stormy night, this seems indeed to be a plausible assumption. Thus, we define our stochastic generalized forces as:

$$Q = (u + \sigma^0 \omega) - b\dot{\theta}, \tag{2.8}$$

where σ^0 represents the magnitude of the noise. Combining and reorganizing equations 2.6, 2.7, and 2.8 results in the following stochastic equations of motion:

$$\begin{aligned} d\theta &= \dot{\theta} dt, \\ d\dot{\theta} &= \frac{(u dt + \sigma^0 d\omega) - (b\dot{\theta} + mgl \sin\theta) dt}{ml^2}. \end{aligned} \tag{2.9}$$

A simulation of such a system is provided in figure 2.5.

Now at the end of this brief treatment of robot kinematics we have so far only discussed low dimensional systems, and it should be asked how relevant this is to robot kinematics in general. The answer is, indeed, very relevant.

As a series of bones and joints, animal skeletons can be modelled more or less as a series of interconnected pendulums. Tendons in turn may be modelled as springs, which themselves are harmonic oscillators. Muscles, finally, are simply where we apply our control. By analyzing high dimensional systems with Lagrangian mechanics, we can generate our high dimensional equations of motion from an analysis involving only these basic components. To officially complete our picture, we first present the following general equations of motion for deterministic robot kinematics, which are sometimes referred to as the *manipulator equations* [11]:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\mathbf{u} - \mathbf{D}(\mathbf{q})\dot{\mathbf{q}}, \tag{2.10}$$

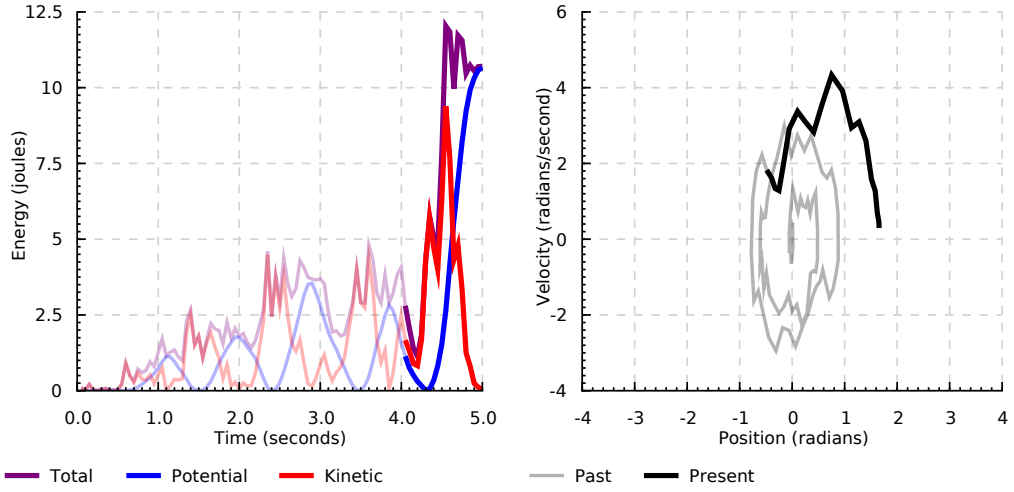

Figure 2.5: **The Stochastic Pendulum**: In this simulation, $u = b = 0$, and $\sigma^0 = 2J$. Left) Initially the pendulum is at rest, but the high degree of noise causes the total energy to take large traversals away from 0. Right) The phase portrait still follows circular trajectories, but in the presence of a high degree of noise.

where $\mathbf{q}$ is the state in generalized coordinates and $\dot{\mathbf{q}}$ is its temporal derivative, $\mathbf{H}$ is the inertial matrix, $\mathbf{C}$ models the coriolis forces and $\mathbf{G}$ the potentials, and finally $\mathbf{B}$ is the control matrix, which modulates the strength of our control at various joints, and $\mathbf{D}$ are the friction and damping terms. When $\mathbf{B}$ is not full rank, the system is referred to as *underactuated*.

These equations are sufficient for modelling a wide variety of control problems, and the control literature is rife with models which take this form. Transforming our manipulator equations into a pair of first order stochastic equations results in the equations of the *manipulator diffusion*:

$$\begin{aligned} d\mathbf{q} &= \dot{\mathbf{q}}dt, \\ d\dot{\mathbf{q}} &= \mathbf{H}^{-1}(\mathbf{q})[\mathbf{B}(\mathbf{q})(\mathbf{u}dt + \sigma^0(\mathbf{q})d\omega) - (\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{D}(\mathbf{q})\dot{\mathbf{q}})dt], \end{aligned} \tag{2.11}$$

where the diffusion σ^0 is square and symmetric, and $d\omega$ is now a multivariate Wiener process.

We conclude with a few remarks. In these equations $\mathbf{B}$ is also multiplied against the Wiener process, which constrains the diffusion to lie within the dimensions of the control. This is the reason for the use of the distinct notation σ^0, since with regards to equation 2.6, $\sigma = \mathbf{B}\sigma^0$.

One may read σ^0 as the 'unconstrained diffusion', and its notation is congruent with the notation developed later in the thesis, where the 0 will indicate variables which are independent of $\mathbf{u}$. Moreover, when needed we will also make use of the notation $\Sigma^0 = \sigma^0\sigma^0$, and $\Sigma = \sigma\sigma^\top = \mathbf{B}\Sigma^0\mathbf{B}^\top$.

Finally, for the sake of brevity, throughout this thesis we will often consider equations of the manipulator diffusion in terms of its complete state vector $\mathbf{x} = [\mathbf{q}, \mathbf{H}\dot{\mathbf{q}}]$, where we multiply the velocities by the inertial matrix in order to simplify the equations over the state vector. Moreover, where n is the dimensional of $\mathbf{Bu}$, $\mathbf{x}$ has dimension $2 \cdot n$. Since we would like to include $\mathbf{B}$ in these equations, assume that in these cases $\mathbf{B}$ represents instead a matrix with $2 \cdot n$ rows, where the first n rows are 0.

Given these assumptions, we may rewrite the manipulator diffusion equation as the controlled Ito diffusion

$$d\mathbf{x} = \mu^0 dt + \mathbf{B}(\mathbf{u}dt + \sigma^0 d\omega), \tag{2.12}$$

where $\mu^0 = \mu - \mathbf{Bu} = [\dot{\mathbf{q}}, \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{D}(\mathbf{q})\dot{\mathbf{q}}]$ is the *uncontrolled drift*. By reversing the aforementioned substitutions, one can take a manipulator diffusion written in the form $\mathbf{x}$, and rewrite it as the coupled diffusion over $\mathbf{q}$ and $\dot{\mathbf{q}}$.

3 Optimal Control

The astute reader will likely have noticed that the oscillatory nature of the simulated system depicted in figure 2.3 is hardly an optimal way to reach the target. The controller should, in the very least, stop applying force at a certain time in order not to overshoot the target. Formally quantifying optimality and subsequently achieving it is the subject of this chapter.

In general what we require is some kind of action policy which describes how to behave under different conditions. In order to solve the problem of calculating an optimal policy, we rely on the well tested and general framework of calculating cost-to-go functions via infinite (time) horizon Bellman equations. We review this approach in section 3.1, beginning with the discrete time control of Markov Decision Processes, and following with the control of continuous time SDEs. Despite its generality, the history of this approach to control has been a mixed success, and the literature is dominated by attempts to wrestle with the theoretical and numerical intractability of the Bellman equations. This practical difficulty of solving the Bellman equations is affectionately known as the 'curse of dimensionality'. Recent work has shown, however, that if we restrict our control problem to one where the cost of control can be characterized as a Kullback Leiber (KL) divergence, we can greatly simplify the Bellman equations. This approach is known as KL Control, and we review it in section 3.2. Finally, through an exponential transformation we can linearize the KL control equations and solve them even more efficiently. We review this final approach in section 3.3.

Each section of this chapter begins by deriving the relevant optimal control theory for Markov Decision Processes, and proceeding it with the theory for controlled stochastic differential equations. In this thesis, we will approximate continuous time problems with discrete time MDPs, and in order to justify the algorithms we develop in the following chapters, we end each section of this one with a convergence lemma justifying our discrete time approximations.

3.1 Bellman Equations

Suppose that we are given an MDP with cost function $l(x,u)$ and transition function $p(x_{i+1} \mid x_i, u_i)$. A policy π is a function from states to actions, and by combining a policy with an MDP we may reduce an MDP to a Markov chain $p^\pi(x_{i+1} \mid x_i) = p(x_{i+1} \mid x_i, \pi(x_i))$. Let us define a policy π as *optimal at* x if it

minimizes the following objective:

$$E^{\pi}(\sum_{i=0}^{\infty} l(x_i, \pi(x_i)) \mid x_0 = x), \tag{3.1}$$

given initial conditions x, where E^{π} is the expected value over the Markov chain defined by p^{π}. Moreover, we define a policy as *globally optimal*, or simply *optimal*, if it is optimal at x for all x. Let us define the *cost-to-go* v^{π} as the solution of the following equation, commonly known as the *Bellman equation*:

$$v^{\pi}(x_i) = l(x_i, \pi(x_i)) + E^{\pi}_{x_{i+1}|x_i}(v^{\pi}(x_{i+1})). \tag{3.2}$$

If we combine equations 3.1 and 3.2, we find that

$$\begin{aligned} v^{\pi}(x_i) &= l(x_i, \pi(x_i)) + E^{\pi}_{x_{i+1} \mid x_i}(v^{\pi}(x_{i+1})) \\ &= E^{\pi}_{x_{i+1} \mid x_i} l(x_i, \pi(x_i)) \mid x_0 = x). \end{aligned}$$

Therefore, if we define *the optimal cost-to-go* v^* as the solution of the *optimal Bellman equation*:

$$v^*(x_i) = \min_u [l(x_i, u) + E^{u}_{x_{i+1} \mid x_i}(v^*(x_{i+1}))], \tag{3.3}$$

then by Bellman's principle of optimality the following greedy policy with respect to the optimal cost-to-go is optimal [10]:

$$\pi^*(x_i) = \arg\min_u [l(x_i, u) + E^{u}_{x_{i+1} \mid x_i}(v^*(x_{i+1}))], \tag{3.4}$$

and finally that:

$$v^{\pi^*} = v^*. \tag{3.5}$$

Thus we have reduced the problem of finding an optimal policy to the problem of computing the optimal cost-to-go.

Let us consider a controlled Ito diffusion $\mathbf{x}$, with vector-valued drift and matrix-valued diffusion functions. Suppose that we are given a continuous time cost, or *loss rate* function l, and that we wish to find a policy π which minimizes the following objective:

$$E^{\pi}(\int_0^{\infty} l(\mathbf{x}_t, \pi(\mathbf{x}_t)) dt \mid \mathbf{x}_0 = x), \tag{3.6}$$

where the expectation E^{π} is now defined over realizations of the controlled diffusion process. Define the *incremental dynamics* of $\mathbf{x}$ as $p^{\pi}(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$, and consider the following differential optimal Bellman equation:

$$v^*(\mathbf{x}_t) = \min_{\mathbf{u}} [l(\mathbf{x}_t, \mathbf{u}) dt + E^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(v^*(\mathbf{x}_{t+dt}))]. \tag{3.7}$$

We would like to reexpress this equation in the limit of $dt \to 0$, so that $v^*(x)$ is the minimum of expression 3.6. Recall that the increments of the underlying Wiener process are normally distributed. Therefore we may rewrite the conditional expectation as:

$$E^{\mathbf{u}}_{\mathbf{x}_{t+dt}|\mathbf{x}_t}(v^*(\mathbf{x}_{t+dt})) = E_{\mathbf{y}\sim\mathbf{N}(0,dt)}(v^*(\mathbf{x}_t + \mu(\mathbf{x}_t,\mathbf{u})dt + \sigma(\mathbf{x}_t,\mathbf{u})\mathbf{y})), \tag{3.8}$$

where $\mathbf{N}(0,dt)$ is a normal distribution with mean 0 and variance $\mathbf{I}dt$. We would now like to approximate this expectation via the Taylor series expansion of v^*. Since v^* is a function of a stochastic process, Ito's lemma suggests that we ought to take the expansion to a higher order in order to capture the higher order moments of the underlying process. Therefore, we calculate the second order expansion of v^* [5, 15]

$$v^*(\mathbf{x}+\mathbf{h}) = v^*(\mathbf{x}) + \mathbf{h}\nabla_{\mathbf{x}}v^*(\mathbf{x}) + \frac{1}{2}\mathbf{h}^\top\nabla_{\mathbf{xx}}v^*(\mathbf{x})\mathbf{h},$$

and by defining $\mathbf{h} = \mu(\mathbf{x}_t,\mathbf{u})dt + \sigma(\mathbf{x}_t,\mathbf{u})\mathbf{y}$, and combining the second order expansion with equations 3.7 and 3.8, we derive

$$\begin{aligned} v^*(\mathbf{x}_t) &= \min_{\mathbf{u}}[l(\mathbf{x}_t,\mathbf{u})dt + E_{\mathbf{y}\sim\mathbf{N}(0,dt)}(v^*(\mathbf{x}_t+\mathbf{h}))] \\ &= \min_{\mathbf{u}}[l(\mathbf{x}_t,\mathbf{u})dt + v^*(\mathbf{x}_t) + E_{\mathbf{y}\sim\mathbf{N}(0,dt)}(\mathbf{h}\nabla_{\mathbf{x}}v^*(\mathbf{x}_t) + \frac{1}{2}\mathbf{h}^\top\nabla_{\mathbf{xx}}v^*(\mathbf{x}_t)\mathbf{h})], \end{aligned}$$

which is equivalent to

$$\begin{aligned} 0 = \min_{\mathbf{u}}[l(\mathbf{x}_t,\mathbf{u})dt + \mu^\top(\mathbf{x}_t,\mathbf{u})\nabla_{\mathbf{x}}v^*(\mathbf{x}_t)dt + \frac{1}{2}\|\mu(\mathbf{x}_t,\mathbf{u})\|^2\nabla_{\mathbf{xx}}v^*(\mathbf{x}_t)dt^2 \\ + \frac{1}{2}E_{\mathbf{y}\sim\mathbf{N}(0,dt)}(\mathbf{y}^\top\sigma(\mathbf{x}_t,\mathbf{u})^\top\nabla_{\mathbf{xx}}v^*(\mathbf{x}_t)\sigma(\mathbf{x}_t,\mathbf{u})\mathbf{y})]. \end{aligned}$$

If we apply the following substitution [14, 12]

$$E(\mathbf{y}^\top\sigma^\top\nabla_{\mathbf{xx}}v^*\sigma\mathbf{y}) = E(\mathrm{Tr}(\sigma\mathbf{y}\mathbf{y}^\top\sigma^\top\nabla_{\mathbf{xx}}v^*)) = \mathrm{Tr}(\Sigma\nabla_{\mathbf{xx}}v^*)dt,$$

and evaluate our derived equation in the limit as $dt \to 0$, we arrive at the general form of the PDE known as the infinite horizon stochastic *Hamilton-Jacobi-Bellman equation* (HJB):

$$0 = \min_{\mathbf{u}}[l(\mathbf{u}) + \mu^\top(\mathbf{u})\nabla_{\mathbf{x}}v^* + \frac{1}{2}\mathrm{Tr}(\Sigma(\mathbf{u})\nabla_{\mathbf{xx}}v^*)]. \tag{3.9}$$

There is a great deal of work in attempting to solve the HJB, and there is a great deal more to be said on the subject. However, in the approach

developed in this thesis, we do not directly rely on the HJB for calculating the optimal cost-to-go. Rather, we compute discrete time approximations of the cost-to-go, and prove that our approximations converge to the solution of the HJB under the right conditions. As regards the topic of this section, that is, Bellman equations in general, we provide the following lemma:

Lemma 3.1. *Let* $\mathbf{x}$ *be a controlled Ito diffusion with lost rate* l *and optimal cost-to-go* v^**, and* $\mathcal{M}_{dt}$ *be the MDP with transition function* $p^\pi(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$ *equal to the incremental dynamics of* $\mathbf{x}$*, cost equal to the discretized loss rate* $l\,dt$*, and optimal cost-to-go* v^*_{dt}*. Then*

$$\lim_{dt \to 0} v^*_{dt} = v^*.$$

Proof. Rewriting equation 3.3 using the given transition function and cost yields 3.7, the solution of which is, as shown, v^* in the limit as $dt \to 0$. ■

3.2 KL Bellman Equations

Let us consider again a discrete time control problem, but this time with the *KL control cost*:

$$l(x,u) = l^0(x) + \mathrm{KL}(p^u(\cdot \mid x) \| p^0(\cdot \mid x)), \tag{3.10}$$

where l^0 is an arbitrary state dependent cost, and where we refer to the transition function $p^0(\cdot \mid x) = p(\cdot \mid x, 0)$ as the *passive dynamics*. Observe that the controlled distribution is free to take any form, constrained only by its cost being a Kullback-Leibler (KL) divergence between the controlled and passive dynamics.

By substituting 3.10 into the optimal Bellman equation we find that

$$\begin{aligned} v^*(x_i) &= \min_u [l^0(x_i) + \mathrm{KL}(p^u(\cdot \mid x_i) \| p^0(\cdot \mid x_i)) + E^u_{x_{i+1} \mid x_i}(v^*(x_{i+1}))], \\ &= l^0(x_i) + \min_u [E^u_{x_{i+1} \mid x_i} \left(\log \frac{p^u(x_{i+1} \mid x_i)}{p^0(x_{i+1} \mid x_i)} + v^*(x_{i+1}) \right)] \\ &= l^0(x_i) + \min_u [E^u_{x_{i+1} \mid x_i} \left(\log \frac{p^u(x_{i+1} \mid x_i)}{p^0(x_{i+1} \mid x_i) \exp(-v^*(x_{i+1}))} \right)]. \end{aligned}$$

Thus, by using the KL control cost we have almost managed to express the second term of our optimal Bellman equation as a minimization over a KL divergence. The only problem remaining is that $p^0(x_{i+1} \mid x_i) \exp(-v^*(x_{i+1}))$ is not normalized. Let us define the *desirability function* as $z = \exp(-v^*)$, along

with the normalizer $Z(x_i) = E^0_{x_{i+1} \mid x_i}(z(x_{i+1}))$. Rewriting and continuing our derivation yields

$$\begin{aligned} v^*(x_i) &= l^0(x_i) + \min_u [E^u_{x_{i+1} \mid x_i} \left(\log \frac{p^u(x_{i+1} \mid x_i)}{p^0(x_{i+1} \mid x_i) z(x_{i+1})} \right)] \\ &= l^0(x_i) + \min_u [E^u_{x_{i+1} \mid x_i} \left(\log \frac{p^u(x_{i+1} \mid x_i) Z(x_i)}{p^0(x_{i+1} \mid x_i) z(x_{i+1}) Z(x_i)} \right)] \\ &= l^0(x_i) - \log Z(x_i) + \min_u \mathrm{KL} \left(p^u(\cdot \mid x_i) \middle\| \frac{p^0(\cdot \mid x_i) z(\cdot)}{Z(x_i)} \right). \end{aligned}$$

The KL divergence achieves its minimum if and only if the argument distributions are equal, and therefore we can directly express the optimally controlled distribution as

$$p^{\pi^*}(x_{i+1} \mid x_i) = \frac{p^0(x_{i+1} \mid x_i) z(x_{i+1})}{Z(x_i)}. \tag{3.11}$$

Finally, the minimum of the KL divergence is 0, and we can therefore write

$$v^*(x_i) = l^0(x_i) - \log Z(x_i) = l^0(x_i) - \log E^0_{x_{i+1} \mid x_i}(\exp(-v^*(x_{i+1}))). \tag{3.12}$$

Thus, we have reduced the equation for the optimal cost-to-go to one purely in terms of the passive dynamics. Therefore, we do not need an explicit model of the effect of control in order to compute the optimal control, yielding a powerful off-policy method. Moreover, in removing the explicit minimization over u we greatly reduce the complexity of various computations in KL control.

Exploring KL control has resulted in a fruitful transformation of the optimal Bellman equation, but in order to compute the optimal policy as per equation 3.4, we would still have to perform a minimization over a potentially infinite space. As it is, equation 3.11 does not yield a policy, but only the optimally controlled distribution. In order to compute the optimal policy, we require more information about the structure of p itself.

Let us temporarily leave behind KL control as such, and consider a manipulator diffusion $\mathbf{x}$. We define our loss rate as the *quadratic loss*:

$$l(\mathbf{x}, \mathbf{u}) = l^0(\mathbf{x}) + \frac{1}{2} \mathbf{u}^\top (\Sigma^0(\mathbf{x}))^{-1} \mathbf{u}, \tag{3.13}$$

where l^0 is a state dependant loss rate. In general, equation 3.13 restricts directions of cheap control to be those in which we have high variance in our underlying white noise process. Since we are free to define l^0, we may simply

scale this term up and down as needed to increase the tendency to act. If we imagine that our larger muscles are subject to greater noise, but less costly to apply, then this restriction makes a degree of intuitive sense.

Inserting this loss rate into the HJB yields

$$\begin{aligned} 0 &= \min_{\mathbf{u}} [l^0 + \frac{1}{2}\mathbf{u}^\top (\Sigma^0)^{-1}\mathbf{u} + \mu^\top(\mathbf{u})\nabla_{\mathbf{x}} v^* + \frac{1}{2}\operatorname{Tr}(\Sigma \nabla_{\mathbf{xx}} v^*)] \\ &= \min_{\mathbf{u}} [l^0 + \frac{1}{2}\mathbf{u}^\top (\Sigma^0)^{-1}\mathbf{u} + (\mu^0)^\top \nabla_{\mathbf{x}} v^* + (\mathbf{B}\mathbf{u})^\top \nabla_{\mathbf{x}} v^* + \frac{1}{2}\operatorname{Tr}(\Sigma \nabla_{\mathbf{xx}} v^*)]. \end{aligned}$$

Notice that $\mathbf{u}$ is now expressed only as a vector. In this quadratic form we can explicitly compute the gradient of the expression inside the minimization, yielding the following optimal policy:

$$\pi^* = -\Sigma^0 \mathbf{B}^\top \nabla_{\mathbf{x}} v^*. \tag{3.14}$$

Inserting this optimal policy into our derivation results in the following simplified HJB:

$$0 = l^0 + (\mu^0)^\top \nabla_{\mathbf{x}} v^* + \frac{1}{2}\operatorname{Tr}(\Sigma \nabla_{\mathbf{xx}} v^*) - \frac{1}{2}(\nabla_{\mathbf{x}} v^*)^\top \Sigma \nabla_{\mathbf{x}} v^*. \tag{3.15}$$

As in the case of our discrete time KL control analysis, our restriction of the loss rate has allowed us to explicitly solve the minimization in the HJB.

Suppose that we wish to approximate the solution of equation 3.15 using a discrete time MDP, and with it the optimal policy of equation 3.14. We know that we can model the incremental dynamics with an MDP, and so the primary question is how to model the quadratic loss. The reader will likely have already guessed that there is a connection between the KL control cost and quadratic loss rate, and indeed we show this in the following lemma:

Lemma 3.2. *Suppose that* $\mathbf{x}$ *is a manipulator diffusion with quadratic loss* l *and state dependent loss rate* l^0*, and optimal cost-to-go* v^**, and let* $\mathscr{M}_{dt}$ *be the MDP with transition function* $p^\pi(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$ *equal to the incremental dynamics of* $\mathbf{x}$*, KL control cost* l_{dt}*, state dependent cost* $l^0 dt$*, and optimal cost-to-go* v^*_{dt}*. Then*

$$l_{dt} = l dt,$$

and therefore

$$\lim_{dt \to 0} v^*_{dt} = v^*.$$

Proof. Observe that $p^{\mathbf{u}}(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$ has normally distributed increments

$$\begin{aligned} p^{\mathbf{u}}(\mathbf{x}_{t+dt} \mid \mathbf{x}_t) &= \mathbf{N}(\mathbf{x}_t + \mu^0(\mathbf{x}_t)dt + \mathbf{B}(\mathbf{x}_t)\mathbf{u}dt, \Sigma(\mathbf{x}_t)dt) \\ p^0(\mathbf{x}_{t+dt} \mid \mathbf{x}_t) &= \mathbf{N}(\mathbf{x}_t + \mu^0(\mathbf{x}_t)dt, \Sigma(\mathbf{x}_t)dt), \end{aligned}$$

where $\mathbf{N}$ is a multivariate normal distribution. In general, the KL divergence between two normal distributions with equal covariance matrices is

$$\mathrm{KL}(N(\mathbf{x},\mathbf{C})\|N(\mathbf{y},\mathbf{C})) = \frac{1}{2}(\mathbf{y}-\mathbf{x})^\top \mathbf{C}^{-1}(\mathbf{y}-\mathbf{x}).$$

Computing the KL control cost we find

$$\begin{aligned}\mathrm{KL}(p^{\mathbf{u}}(\cdot \mid \mathbf{x}_t)\|p^0(\cdot \mid \mathbf{x}_t)) &= \frac{1}{2}(\mathbf{B}(\mathbf{x}_t)\mathbf{u})^\top \Sigma^{-1}(\mathbf{x}_t)\mathbf{B}(\mathbf{x}_t)\mathbf{u}dt \\ &= \frac{1}{2}\mathbf{u}^\top(\Sigma^0(\mathbf{x}_t))^{-1}\mathbf{u}dt\end{aligned}$$

which is equivalent to the fact that

$$l_{dt} = l^0 dt + \mathrm{KL}(p^{\mathbf{u}}(\cdot \mid \mathbf{x}_t)\|p^0(\cdot \mid \mathbf{x}_t)) = l^0 dt + \frac{1}{2}\mathbf{u}^\top(\Sigma^0(\mathbf{x}_t))^{-1}\mathbf{u}dt = l dt.$$

The second equation then follows from lemma 3.1. ■

By approximating the optimal-cost-to-go of the manipulator diffusion with that of an MDP with KL control cost, we may calculate the optimal policy as desired.

3.3 Linear Bellman Equations

KL control has an additional interesting property, namely, that we can linearize the optimal Bellman equation by negating and exponentiating both sides:

$$\begin{aligned} & v^*(x_i) = l^0(x_i) - \log E^0_{x_{i+1} \mid x_i}(\exp(-v^*(x_{i+1}))) \\ \iff\ & z(x_i) = \exp(-l^0(x_i))Z(x_i) \\ \iff\ & z = \exp(-l^0)Z(z), \end{aligned} \tag{3.16}$$

where we explicitly express the dependence of Z upon z in equation 3.16. Since expectations are linear operators, the normalizer Z is linear in z, resulting in a linear equation of the optimal desirability.

We can also linearize the HJB of the manipulator diffusion by again using the desirability function. Equating the derivatives of the cost-to-go with those of the desirability function results in [12]

$$\nabla_{\mathbf{x}} v^* = -\frac{\nabla_{\mathbf{x}} z}{z}, \quad \nabla_{\mathbf{xx}} v^* = \frac{\nabla_{\mathbf{x}} z \nabla_{\mathbf{x}} z^\top}{z^2} - \frac{\nabla_{\mathbf{xx}} z}{z}. \tag{3.17}$$

Applying these substitutions to equation 3.15, and relying on certain properties of the trace operator yields [15]

$$\begin{aligned} 0 &= l^0 - \frac{1}{z}\big((\mu^0)^\top \nabla_{\mathbf{x}} z + \frac{1}{2}\operatorname{Tr}(\Sigma \nabla_{\mathbf{xx}} z) + \frac{\nabla_{\mathbf{x}} z^\top \Sigma \nabla_{\mathbf{x}} z - \nabla_{\mathbf{x}} z^\top \Sigma \nabla_{\mathbf{x}} z}{2z}\big). \\ &= l^0 - \frac{1}{z}\big((\mu^0)^\top \nabla_{\mathbf{x}} z + \frac{1}{2}\operatorname{Tr}(\Sigma \nabla_{\mathbf{xx}} z)\big), \end{aligned} \tag{3.18}$$

which is a linear PDE, the solution of which is the optimal desirability. We can express the optimal policy as a function of the desirability by simply substituting the gradient from 3.17 into equation 3.14 resulting in

$$\pi^* = \Sigma^0 \mathbf{B}^\top \frac{\nabla_{\mathbf{x}} z}{z}. \tag{3.19}$$

For all its beauty, equation 3.18 is still not trivial to solve. The so called path integral approach to control relies on the Feynman-Kac lemma to transform a similar equation into a more computationally tractable path integral, and research in this area is ongoing [6, 13, 8]. However, as before, in this thesis we will rely on discrete time, linearized KL control to efficiently compute an approximation of our optimal desirability via equation 3.16, from which we can compute the optimal policy using equation 3.19. In closing this chapter, we provide our last convergence lemma.

Lemma 3.3. *Suppose that* $\mathbf{x}$ *is a manipulator diffusion with quadratic loss* l *and state dependent loss rate* l^0*, and optimal desirability* z*. Let* $\mathcal{M}_{dt}$ *be the MDP with transition function* $p^\pi(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$ *equal to the incremental dynamics of* $\mathbf{x}$*, KL control cost* l_{dt}*, state dependent cost* $l^0 dt$*, and optimal desirability* z*. Then*

$$l_{dt} = l dt,$$

and therefore

$$\lim_{dt \to 0} z_{dt} = z.$$

Proof. Let $p^u(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$ be the discrete time transition function of the SDE defined by equation 2.12, and let l be the quadratic loss rate. We may then define the discretized state dependant cost function as $l^0 dt$, and beginning with the linear Bellman equation derive

$$\begin{aligned} & z_{dt}(\mathbf{x}_t) = e^{-dt l^0(\mathbf{x}_t)} E^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(z_{dt}(\mathbf{x}_{dt+t})) \\ \iff \quad & z_{dt}(\mathbf{x}_t) e^{dt l^0(\mathbf{x}_t)} - z_{dt}(\mathbf{x}_{dt}) = E^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(z_{dt}(\mathbf{x}_{dt+t})) - z_{dt}(\mathbf{x}_{dt}) \\ \iff \quad & \frac{z_{dt}(\mathbf{x}_t)(e^{dt l^0(\mathbf{x}_t)} - 1)}{dt} = \frac{E^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(z_{dt}(\mathbf{x}_{dt+t}) - z_{dt}(\mathbf{x}_{dt}))}{dt}. \end{aligned} \tag{3.20}$$

Let us assume that $\lim_{dt\to 0} z_{dt} = z$ for some as of yet unknown function z'. By applying the same methods used to analyze the conditional expectation in equation 3.8, we can show that:

$$\lim_{dt\to 0} \frac{E^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(z_{dt}(\mathbf{x}_{dt+t}) - z_{dt}(\mathbf{x}_{dt}))}{dt} = (\mu^0)^\top \nabla_{\mathbf{x}} z' + \mathrm{Tr}(\frac{1}{2}\Sigma \nabla_{\mathbf{xx}} z') \tag{3.21}$$

By taking the limit of both sides of equation 3.20, and applying 3.21 as well as basic theorems of calculus we derive [15, 7]

$$\begin{aligned} & z' l^0 = (\mu^0)^\top \nabla_{\mathbf{x}} z' + \mathrm{Tr}(\frac{1}{2}\Sigma \nabla_{\mathbf{xx}} z) \\ \iff \quad & 0 = l^0 - \frac{1}{z'}((\mu^0)^\top \nabla_{\mathbf{x}} z' + \mathrm{Tr}(\frac{1}{2}\Sigma \nabla_{\mathbf{xx}} z')), \end{aligned}$$

which is the linear HJB. Therefore, assuming our discretized desirability z_{dt} is well behaved, it converges to the solution of the linear HJB of the manipulator diffusion, so that $z' = z$. ■

4 Kernel Methods

Conditional expectations are ubiquitous in chapter 3, where they serve to encode the dynamics of the underlying system in the Bellman equations. Chapter 3 was driven by a desire to find efficient expressions for both the optimal cost-to-go and policy of stochastic systems, but even where we have managed to analyze away minimizations and linearize equations, it will be all for nought unless we can evaluate the conditional expectations.

Given that we will often have explicit equations of motion of our dynamics, incorporating this information into our evaluations would be at least one step towards solving this problem. However, driven by a desire for adaptive, realistic control, we do not wish to rely on built in models of system dynamics in order for a robot to evaluate these expectations. Rather, we wish for the robot to build a model from experience.

Recent work in kernel methods has yielded a general purpose framework for sample based approximations of conditional expectations which, at least theoretically, satisfy our requirements to the letter. In section 4.1 we will develop this framework of embedding probability distributions in reproducing kernel Hilbert spaces. In section 4.2 we will show how these embeddings can be applied to modelling the incremental dynamics of a controlled Ito diffusion.

4.1 RKHS Embeddings

Suppose that we are given a set $\mathcal{X}$, and a symmetric kernel, $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which is *positive definite*:

$$\sum_{i,j=1}^{n} c_i c_j k(x_i, x_j) \geq 0, \forall \{x_1, \ldots, x_n\} \subset \mathcal{X}, \forall \{c_1, \ldots, c_n\} \subset \mathbb{R}. \tag{4.1}$$

Then k implicitly defines a Hilbert space $\mathcal{H} = \text{Span}\{k(\cdot, x) \mid x \in \mathcal{X}\}$ with inner product:

$$\langle f, g \rangle = \sum_{i,j=1}^{n} a_i b_j k(x_i, x_j), \forall f, g \in \mathcal{H}, \tag{4.2}$$

where a_i, b_j are the weights which define f and g in the span of the kernel basis. Moreover, in this Hilbert space, k has the *reproducing property*:

$$\langle f, k(x, \cdot) \rangle = f(x), \forall f \in \mathcal{H}. \tag{4.3}$$

leading to the definition of $\mathscr{H}$ as a *reproducing kernel Hilbert space* (RKHS).

Suppose that we are also given a probability distribution P on $\mathscr{X}$ with associated expectation operator E. Then we may define the *kernel mean* as $m_P = m_X = E_{X\sim P}(k(X,\cdot)) \in \mathscr{H}$. Moreover, by the reproducing property:

$$\langle f, m_P \rangle = E_P(f), \forall f \in \mathscr{H}. \tag{4.4}$$

Since expectations are intuitively simply the integral over the product of a function and a distribution, we may consider m_P as an embedding of P in $\mathscr{H}$. Now in general computing expectations over functions is computationally intractable. Nevertheless, given a sample $\{X_1,\ldots,X_n\} \sim P$, this definition yields the following trivial empirical estimator:

$$\hat{m}_P(x) = \frac{1}{n}\sum_{i=1}^{n} k(X_i, x) \tag{4.5}$$

Thus, for all the theory so far, we are in fact doing something very simple. If the kernel k is normalized to integrate to 1, then $\hat{m}_P$ is simply the kernel density estimate of our sample (figure 4.1). Of course, if this were all we could do with RKHS embeddings it would not be sufficient for our purposes, and the real innovation of this approach comes from its ability to approximate conditional distributions $P_{Y|X}$.

Suppose we now have have a joint distribution P over $(\mathscr{X},\mathscr{Y})$, with associated kernels k_X and k_Y, and Hilbert spaces $\mathscr{H}_X$ and $\mathscr{H}_Y$. We implicitly define

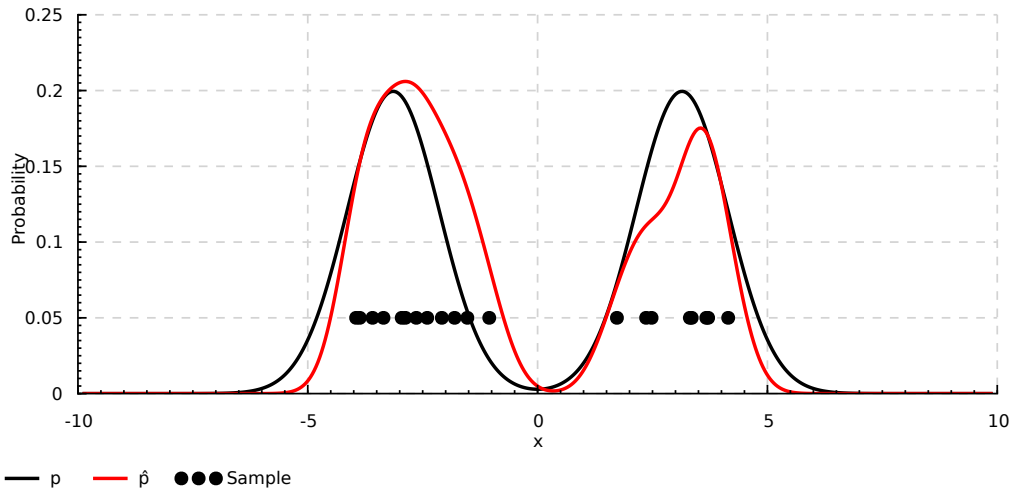

Figure 4.1: **Kernel Density Estimation**: An example of kernel density estimation over $\mathscr{X} = \mathbb{R}$ with a Gaussian kernel of width $w = 0.5$, and depicted probability density p. The sample size is $n = 20$ (NB: the ordinate of the sample points is meaningless).

the (uncentred) *covariance operator* as the function $C_{YX} : \mathscr{H}_X \to \mathscr{H}_Y$ which satisfies:

$$\langle g, C_{YX} f \rangle_{k_Y} = E_P(f(X)g(Y)), \forall f \in \mathscr{H}_X, g \in \mathscr{H}_Y. \tag{4.6}$$

Covariances operators characterize the joint distribution between two reproducing kernel Hilbert spaces, and are our key for modelling conditional distributions. Now, for a given x, $P_{Y|x}$ is a distribution over $\mathscr{Y}$. What we would therefore like to do is directly embed this conditional distribution into $\mathscr{H}_Y$ as the kernel mean $m_{Y|x}$. However, the covariance operator C_{YX} only relates pairs of functions from RKHSs. Individual points cannot live in an RKHS, and therefore covariance operators cannot relate a distribution with a point.

Nevertheless, what can be shown is that covariance operators can be used to compute marginal distributions of the form $Q = \int P_{Y|x}\Pi(x)dx$. If we define $k_\Pi = k_X(x, \cdot)$ for a given x, then k_Π can be interpreted as a point approximation to x in $\mathscr{H}_X$, and Q therefore as an approximation to $P_{Y|x}$. This approach is driven by the following theorem:

Theorem 4.1. *Suppose that m_Π and m_Q are the kernel means of Π in $\mathscr{H}_X$ and Q in $\mathscr{H}_Y$, respectively. If C_{XX} is invertible, $m_\Pi \in \text{Range}(C_{XX})$, and $E(g(Y) \mid X = \cdot) \in \mathscr{H}_X$, then [9, 2]*

$$m_Q = C_{YX} C_{XX}^{-1} m_\Pi$$

An appropriate empirical estimator of $m_{P_{Y|x}}$ is not as immediately forthcoming as that provided in equation 4.5. In order to quantify the quality of a conditional mean embedding, we consider the following objective:

$$\sup_{\|f\|_Y \leq 1} E_X[E_Y(f(Y) \mid X) - \langle f, m_{P_{Y|X}}(X) \rangle_{k_Y}]. \tag{4.7}$$

This definition allows us to state the following theorem:

Theorem 4.2. *Define $m_{P_{Y|x}} = C_{YX} C_{XX}^{-1} k_X(x, \cdot)$, and let $\{(X, Y)_1, \ldots, (X, Y)_n\} \sim P$, $\mathbf{K}_{X,(i,j)} = k_X(X_i, X_j)$ denote the Gramian matrix with respect to k_X, $\mathbf{k}_{X,(i)}(x) = k(X_i, x)$ be the representation of the point x in terms of the sample basis, and λ be a regularization parameter. Then an empirical estimate which minimizes an approximate, regularized bound on 4.7 with respect to $m_{P_{Y|x}}$ is [9, 4]*

$$\hat{m}_{P_{Y|x}} = \sum_{i=1}^{n} \alpha_i(x) k_Y(Y_i, \cdot),$$

where

$$\mathbf{W} = (\mathbf{K}_X + \lambda n \mathbf{I})^{-1},$$

and

$$\alpha(x) = \mathbf{W} \cdot \mathbf{k}_X(x).$$

Given a sample $\{(X,Y)_1,\ldots,(X,Y)_n\} \sim P$, we may then define our sample based approximation of the conditional expectation over $P_{Y|x}$ for a given x as:

$$\hat{E}_{Y|x}(f) = \langle f, \hat{m}_{P_{Y|x}} \rangle = \mathbf{f}_X^\top (\mathbf{K}_X + \lambda n \mathbf{I})^{-1} \mathbf{k}_X(x), \tag{4.8}$$

where $\mathbf{f}_{X,(i)} = f(X_i)$. This is the primary result that we have been after.

We are now ready to begin applying kernel means to the modelling of the conditional expectations in the Bellman equations. However, before we close this section, we will discuss a few applications of RKHS embeddings in order to demonstrate their power and range.

Firstly, we provide a simple example of function regression using kernel means in figure 4.2. Calculating the various moments of the regression, i.e. the mean and standard deviation, simply involves first estimating $\hat{E}_{Y|x}$ for every x at a desired resolution, and then evaluating our quantities of interest at each x. To reiterate, the estimation of $\hat{E}_{Y|x}$ is independent of the function we wish to integrate, and therefore we may reuse our estimated expectation operators as often as we wish. Moreover, once we have computed α, evaluating $\hat{E}_{Y|x}$ is of complexity $O(n)$ in the sample size n. Expectations are the basis for a wide range of statistical and information theoretic quantities, and the methods of RKHS embeddings is therefore indeed applicable to a wide range of tasks.

Readers familiar with Gaussian process (GP) regression may have noticed the striking similarity between the algorithms underlying kernel means and GP

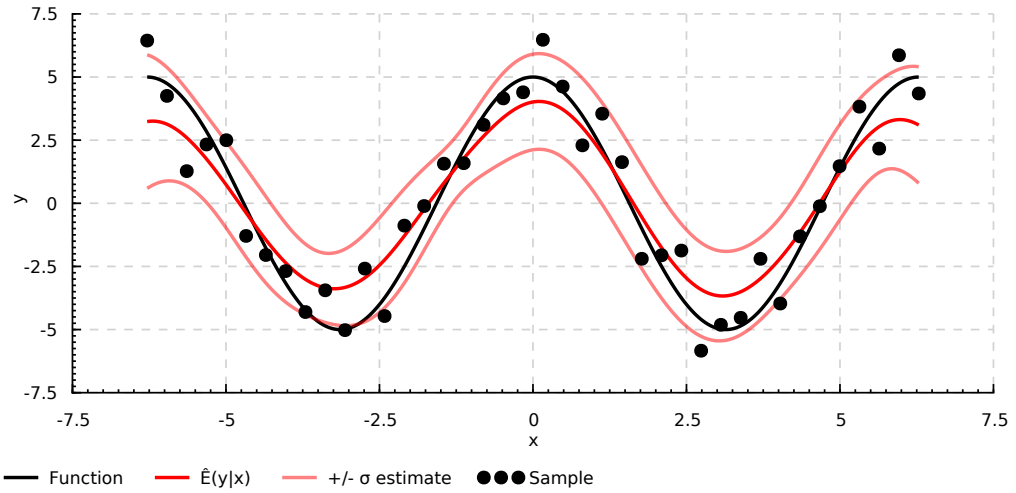

Figure 4.2: **Kernel Mean Regression**: Regression on a sinusoid based on $n = 40$ observations with additive standard normal noise. k_x is a Gaussian kernel of width $w = 0.8$, and the regularization parameter $\lambda = 0.03$. Since the standard deviation is simply another expectation, we may estimate it as well at every x.

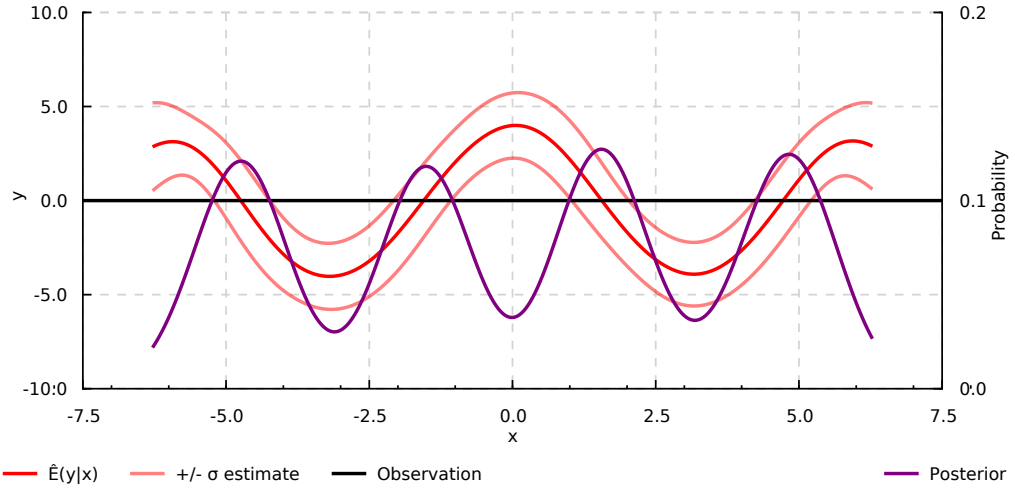

Figure 4.3: **Kernel Bayes' Rule**: We estimate the likelihood, again plotted in red, with the same parameters and underlying function as in figure 4.2, with the exception that $n = 400$ as more data is required to achieve good results. Given the likelihood and a flat prior, we make the observation that $y = 0$, and compute the posterior $q(x \mid 0)$. The result, plotted against the right vertical axis, is a distribution with probability mass concentrated around those points where we expect to find x given that $y = 0$.

regression. In fact, by taking the right hand side of equation 4.8, transposing, and substituting $\sigma^2 = \lambda n$ we recover the standard form of the equivalent GP regression formula. As far as the applications of this thesis are concerned, we could have indeed used the GP regression framework instead.

The theoretical bases of GP regression and RKHS embeddings are nevertheless not the same, and both offer certain advantages over the other. That being said, computationally speaking, GP regressions are a special case of RKHS embeddings. Since the result of theorem 4.2 is a general purpose kernel mean, we may apply further tools to this element of $\mathscr{H}_Y$ beyond only evaluating a conditional expectation, as is the case with GP regression. We close this section with one such example, namely in providing an RKHS embedding of Bayes' rule [2].

While we will not go into the details of the algorithm, from an intuitive perspective, assuming that we can compute our likelihood kernel means $m_{P_{Y|x}}, \forall x$, we need to integrate it with our prior m_Π and normalize it. Theorem 4.1 provides the first step in this calculation, and through another matrix inversion we may compute the normalizer and finally the posterior distribution $m_{Q_{X|y}}$. We plot the results of such a calculation in figure 4.3.

Algorithm 1 Modelling the Incremental Dynamics

Require: $\pi, \mathscr{S}^{\pi}_{dt}, k_{(x,u)}, k_x, \lambda$
$\mathbf{W} \leftarrow (\mathbf{K}_{(x,u)} + \lambda n \mathbf{I})^{-1}$
return $\hat{m}^{\pi}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t} = \mathbf{k}_{\mathbf{x}}^{\top} \cdot \mathbf{W} \cdot \mathbf{k}_{(\mathbf{x},\mathbf{u})}(\mathbf{x}_t, \pi(\mathbf{x}_t))$

4.2 Modelling Incremental Dynamics

Let us now embed the incremental dynamics of an Ito diffusion into an RKHS, so that we may estimate the solutions of the various Bellman equations that were developed in chapter 3. Suppose the following:

- $\mathbf{x}$ is a controlled Ito diffusion with incremental dynamics $p^{\pi}(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$, where $\mathscr{X}$ is the phase space of $\mathbf{x}$ and $\mathscr{U}$ is the set of possible actions $\mathbf{u}$.
- $k_{(\mathbf{x},\mathbf{u})}$ is a kernel over $(\mathscr{X}, \mathscr{U})$ with RKHS $\mathscr{H}_{(\mathbf{x},\mathbf{u})}$, and $k_{\mathbf{x}}$ is a kernel over $\mathscr{X}$ with RKHS $\mathscr{H}_{\mathbf{x}}$.
- $\mathscr{S}^{\pi}_{dt} = \{((\mathbf{x}, \pi(\mathbf{x})), \mathbf{x}')_1, \ldots, ((\mathbf{x}, \pi(\mathbf{x})), \mathbf{x}')_n\} \sim p^{\pi}(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$ is a set of sample transitions from the incremental dynamics at a given temporal resolution dt.

Then we may compute the empirical RKHS embedding $\hat{m}^{\pi}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}$ with theorem 4.2 as summarized in algorithm 1. With our estimated embedding we may then calculate the estimated conditional expectation operator $\hat{E}^{\pi}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}$ using equation 4.8.

Although this algorithm requires no information about the parameters of the underlying diffusion, in order to build a successful model we must still tune the kernels $k_{\mathbf{x}}$ and $k_{(\mathbf{x},\mathbf{u})}$, and the regularization parameter λ. Given a test set of transitions $\mathscr{T}^{\pi}_{dt} = \{((\mathbf{x}, \pi(\mathbf{x})), \mathbf{x}')_1, \ldots, ((\mathbf{x}, \pi(\mathbf{x})), \mathbf{x}')_m\} \sim p^{\pi}(\mathbf{x}_{t+dt} \mid \mathbf{x}_t)$, we tune the parameters using cross-validation, by calculating model scores based on the following empirical error function [4]:

$$\sum_{i=1}^{m} \| k_{\mathbf{x}}(\mathbf{x}'_i, \cdot) - \mathbf{k}_{\mathbf{x}} \cdot \mathbf{W} \cdot \mathbf{k}_{(\mathbf{x},\mathbf{u})}(\mathbf{x}_i, \pi(\mathbf{x}_i)) \|_{k_{\mathbf{x}}}, \tag{4.9}$$

which can be shown to be an approximate bound on objective 4.7.

As stated in theorem 4.2, the empirical estimator in algorithm 1 minimizes a regularized version of objective 4.7. The regularization term is of the form $\lambda \|m\|^2$, where the norm is over the magnitude of the weights of our RKHS embedding m. λ is therefore an L^2 regularization parameter, which both prevents overfitting and provides numerical stability. As λ increases, so does the

relative cost of large weights in the resulting embedding, thus encouraging smaller, more stable weights. If $\lambda = 0$, the presented algorithms will usually fail to return a consistent solution. For more details see [2, 4].

With regards to kernel selection, although we also ran experiments with wavelet kernels as presented in [16], in general we found that *Gaussian kernels* were sufficient for our purposes:

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{1}{2}\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{w}\right)^2}, \tag{4.10}$$

where w is the *width* of the kernel.

As such, we set ourselves to modelling the incremental dynamics of an Ito diffusion $\mathbf{x}$, by tuning the parameters w and λ, based on cross-validation scores from expression 4.9. In order to validate this approach, let us estimate drifts $\hat{\mu}_{w,\lambda}$ based on samples $\mathscr{S}^0_{dt}$ from an uncontrolled, stochastic pendulum and compare it to the true drift μ. We may then compare the performance of the $\hat{\mu}_{w,\lambda}$ in this drift modelling task with their calculated cross-validation scores.

Given an estimation of the incremental expectation $\hat{E}^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}$, we can easily approximate the drift of the underlying diffusion as

$$\begin{aligned} & \hat{E}^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\mathbf{x}_{t+dt}) = \hat{\mu}_{w,\lambda}(\mathbf{x}_t)dt \\ \Leftrightarrow \quad & \hat{E}^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\mathbf{x}_{t+dt})/dt = \hat{\mu}_{w,\lambda}(\mathbf{x}_t). \end{aligned} \tag{4.11}$$

Recall that in the context of stochastic equations of motion, μ represents the deterministic dynamics of the underlying system. Therefore, a natural way to visualize the correspondence between the true μ and estimators $\hat{\mu}_{w,\lambda}$ is by comparing their vector fields. We provide the results of such experiments in figure 4.4.

As figure 4.4 demonstrates, the error function 4.9 seems to measure the quality of our estimators quite well. Notice that larger kernel widths appear to provide better performance. This comes however at the cost of requiring less regularization λ, which will tend to lead to greater numerical instability. Therefore, based on these results we would rely on the estimator $\hat{\mu}_{w,\lambda}$ with parameters $w = 4.0$ and $\lambda = 10^{-6}$ for an 9×9 grid of samples $\mathscr{S}_{dt}$. We will rely on such results as the basis for our simulations and control computations in the following chapters.

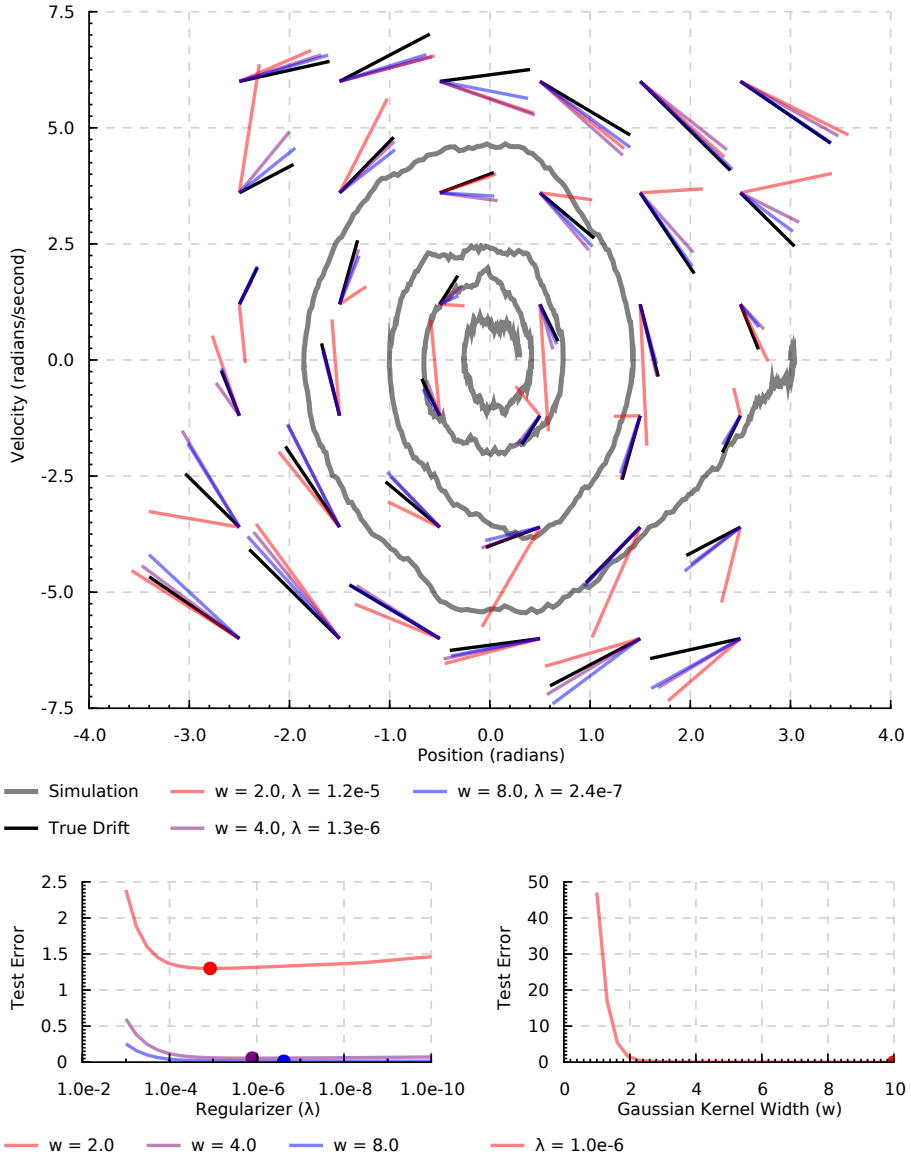

Figure 4.4: **Modelling a Vector Field**: We simulate the dynamics of an uncontrolled stochastic pendulum with model parameters $\sigma = 0.5J$ and $b = 0.5 \cdot m \cdot kg/s$ (otherwise as per figure 2.2), at temporal resolution $dt = 0.01$. $\mathscr{S}^0_{dt}$ is an 9×9 grid of samples from the incremental dynamics, and $\mathscr{T}^0_{dt}$ is a 16×16 grid of such samples.
Top) Estimated drifts $\hat{\mu}_{a,\lambda}$ for kernel width a, and regularization parameter λ, compared against the true drift μ. The blue and purple drift models perform well, and are generally close to the black (true) drift. In comparison the red drift model performs poorly.
Bottom) The error 4.9 on $\mathscr{T}^0_{dt}$. *Left)* The error as a function of λ. The minima are achieved at the coloured dots. *Right)* The error as a function of a. Large kernel widths are advantageous.
Results: The parameters $w = 4.0$ and $\lambda = 10^{-6}$ provide a good balance between performance and numerical stability.

5 Kernelized Optimal Control

Ready with the tools to estimate our conditional expectations, we must now develop the theory of chapter 3 into a set of algorithms for approximating solutions to the Bellman equations, with the overall goal of solving the continuous time control problems of chapter 2. These algorithms will take the approach of approximating the solution of a discrete time Bellman equation, for an MDP which models the incremental dynamics of a controlled Ito diffusion.

Consider again equation 3.9, the infinite horizon Hamilton-Jacobi-Bellman equation. Intuitively its solution, the optimal cost-to-go, gathers all available information about future costs into the current cost. On reflection, it is clear that over the course of an infinite future, there is a potentially infinite amount of cost to be accrued. Therefore, it is not generally the case that the HJB will have a bounded solution.

Thus, in practical approaches to infinite horizon problems we reformulate the objective of expression 3.6 in one of two ways. The first reformulated objective is known as the *infinite horizon discounted cost*, and the second as the *infinite horizon average cost*. We review algorithms for the estimation and approximation of discounted cost problems in section 5.1, and of average cost problems in section 5.2.

In chapter 3 we presented three forms of the HJB with accompanying discrete time approximation lemmas. The form of the approximating Bellman equations results in algorithms which are more are less appropriate for discounted cost or average cost problems. As we will see, the general methods of section 3.1 and the KL control methods of section 3.2 will work best in the context of discounted cost problems, and the linearized methods of section 3.3 will be better suited to solving average cost problems.

5.1 Kernelized Value Iteration

In discounted cost problems, the *discount factor* $0 < \gamma < 1$ serves to recursively scale down future costs. For a controlled Ito diffusion, the reformulated objective is

$$E^{\pi}\left(\int_0^{\infty} e^{-\gamma t} l(\mathbf{x}_t, \pi(\mathbf{x}_t)) dt \mid \mathbf{x}_0 = x\right), \tag{5.1}$$

resulting in the following discounted cost HJB [14, 15]:

$$\gamma v^* = \min_{\mathbf{u}}[l(\mathbf{u}) + \mu^\top(\mathbf{u})\nabla_{\mathbf{x}} v^* + \frac{1}{2}\mathrm{Tr}(\sigma^2(\mathbf{u})\nabla_{\mathbf{xx}} v^*)]. \tag{5.2}$$

In order to approximate the solution of equation 5.2 we rely on lemma 3.1. In the discounted cost case, the differential optimal Bellman equation 3.7 becomes

$$v^*(\mathbf{x}_t) = \min_{\mathbf{u}}[l(\mathbf{x}_t, \mathbf{u})dt + e^{-\gamma dt} E^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(v^*(\mathbf{x}_{t+dt}))]. \tag{5.3}$$

One of the most powerful approaches to approximating the solution of a Bellman equation is the method of *value iteration*. This method involves turning the optimal Bellman equation 3.3 into the following recurrence relation for computing the optimal cost-to-go:

$$v^*_{k+1}(x_i) = \min_{u}[l(x_i, u) + E^{u}_{x_{i+1} \mid x_i}(v^*_k(x_{i+1}))], \tag{5.4}$$

where we compute each new iteration v^*_{k+1} as a function of the current iteration v^*_k, by minimizing with respect to u for every x_i. If we apply this approach to our discounted cost, differential Bellman equation, we derive the *differential value iteration* recurrence:

$$v^*_{k+1}(\mathbf{x}_t) = \min_{\mathbf{u}}[l(\mathbf{x}_t, \mathbf{u})dt + e^{-\gamma dt} E^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(v^*_k(\mathbf{x}_{t+dt}))]. \tag{5.5}$$

It is easy to show that, due to the discount factor, this recurrence relation forms a contraction mapping, and therefore that by Banach's fixed point theorem, equation 5.5 converges geometrically to v^* [10].

If we combine this equation with our ability to estimate conditional expectations, we can estimate a solution of the discounted cost HJB 5.2 by iterating the recurrence relation:

$$\hat{v}^*_{k+1}(\mathbf{x}_t) = \min_{\mathbf{u}}[l(\mathbf{x}_t, \mathbf{u})dt + e^{-\gamma dt} \hat{E}^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\hat{v}^*_k(\mathbf{x}_{t+dt}))]. \tag{5.6}$$

In computing this iteration, we can precompute the weights of the necessary RKHS embeddings in time $O((mn)^3)$, thus reducing each evaluation of a conditional expectation to a vector product of time complexity $O(mn)$. We now very nearly have a tractable algorithm for estimating the optimal cost-to-go. The remaining problem is the computation of the minimization over $\mathbf{u}$.

Suppose $|\mathscr{U}| = m < \infty$, and $\mathscr{S}^{\mathbf{u}}_{dt}$ is a set of sample transitions $\{((\mathbf{x}, \mathbf{u}), \mathbf{x}')_i\} \sim p^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}$ such that $\mathbf{u} \in \mathscr{U}$. Then we may solve the minimization by directly calculating the right hand side for each $\mathbf{u} \in \mathscr{U}$, yielding the following computable recurrence relation for estimating the optimal cost-to-go [3]:

$$\hat{v}^*_{k+1}(\mathbf{x}_t) = \min_{\mathbf{u} \in \mathscr{U}}[l(\mathbf{x}_t, \mathbf{u})dt + e^{-\gamma dt} \hat{E}^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\hat{v}^*_k(\mathbf{x}_{t+dt}))]. \tag{5.7}$$

Algorithm 2 Differential Value Iteration

Require: $\gamma, l, \mathscr{U}, \mathscr{S}^{\mathbf{u}}_{dt}, k_{(x,u)}, k_x, \lambda$
  $\mathbf{W} \leftarrow (\mathbf{K}_{(x,u)} + \lambda n \mathbf{I})^{-1}$
  for all $\mathbf{u} \in \mathscr{U}, \mathbf{x}' \in \mathscr{S}^{\mathbf{u}}_{dt}$ **do**
    $\alpha(\mathbf{x}', \mathbf{u}) \leftarrow \mathbf{W} \cdot \mathbf{k}_{(\mathbf{x},\mathbf{u})}(\mathbf{x}', \mathbf{u})$
  end for
  $\mathbf{v}_0 \leftarrow \mathbf{0}$
  repeat
    for all $\mathbf{u} \in \mathscr{U}, \mathbf{x}' \in \mathscr{S}^{\mathbf{u}}_{dt}$ **do**
      $v_{k+1}(\mathbf{x}') \leftarrow \min_{\mathbf{u}\in\mathscr{U}}[l(\mathbf{x}', \mathbf{u})dt + e^{-\gamma dt}\mathbf{v}_k^\top \alpha(\mathbf{x}', \mathbf{u})]$
    end for
  until CONVERGENCE$(\mathbf{v}_{k+1}, \mathbf{v}_k)$
  $\mathbf{v} \leftarrow \mathbf{v}_{k+1}$
  return $\hat{v}^*(\mathbf{x}) = \min_{\mathbf{u}\in\mathscr{U}}[l(\mathbf{x}, \mathbf{u})dt + e^{-\gamma dt}\mathbf{v}^\top \alpha(\mathbf{x}, \mathbf{u})]$

Given the precomputation of the weights, the time complexity of each iteration of the algorithm is $O((mn)^2)$. By iterating the algorithm until convergence, we estimate the optimal cost-to-go $\hat{v}^*$, as summarized in algorithm 2.

With an estimation of the optimal cost-to-go in hand, we may then compute the following estimate of the optimal policy:

$$\hat{\pi}^*(\mathbf{x}_t) = \arg\min_{\mathbf{u}\in\mathscr{U}}[l(\mathbf{x}_t, \mathbf{u})dt + e^{-\gamma dt}\hat{E}^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\hat{v}^*(\mathbf{x}_{t+dt}))]. \tag{5.8}$$

Suppose we do not wish to restrict the size of $\mathscr{U}$, but that we are willing to restrict ourselves to the affine manipulator diffusions. We may then define our loss rate as the quadratic loss, thereby allowing us to rewrite equation 5.6 as

$$\hat{v}^*_{k+1}(\mathbf{x}_t) = \min_{\mathbf{u}}[l^0 + \frac{1}{2}\mathbf{u}^\top(\Sigma^0(\mathbf{x}_t))^{-1}\mathbf{u}dt + e^{-\gamma dt}\hat{E}^{\mathbf{u}}_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\hat{v}^*_k(\mathbf{x}_{t+dt}))].$$

If we then apply theorem 3.2, and carry the term $e^{-\gamma dt}$ through the derivation of equation 3.12, we can again solve the minimization, resulting in the following *KL value iteration* equation:

$$\hat{v}^*_{k+1}(\mathbf{x}_t) = l^0(\mathbf{x}_t)dt - \log \hat{E}^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\exp(e^{-\gamma dt}\hat{v}^*_k(\mathbf{x}_{t+dt}))). \tag{5.9}$$

We summarize this approach to estimating the cost-to-go in algorithm 3. Observe that in developing this algorithm, another advantage of the KL control approach becomes apparent. Since we no longer need to calculate a minimization over some finite set $\mathscr{U}$, the time to precompute the weights is now simply

Algorithm 3 KL Value Iteration

Require: $\gamma, l^0, \mathscr{S}_{dt}^0, k_{\mathbf{x}}, \lambda$
  $\mathbf{W} \leftarrow (\mathbf{K}_{\mathbf{x}} + \lambda n \mathbf{I})^{-1}$
  $\mathbf{K}_{\mathbf{x}\mathbf{x}'} \leftarrow [\mathbf{k}_{\mathbf{x}}(\mathbf{x}'_1), .., \mathbf{k}_{\mathbf{x}}(\mathbf{x}'_n)]$
  $\mathbf{l}^0 \leftarrow [l^0(\mathbf{x}'_1), .., l^0(\mathbf{x}'_n)]^\top$
  $\mathbf{v}_0 \leftarrow \mathbf{0}$
  repeat
    $\mathbf{v}_{k+1} \leftarrow \mathbf{l}^0 dt - \log(\exp(e^{-\gamma dt}\mathbf{v}_k^\top) \cdot \mathbf{W} \cdot \mathbf{K}_{\mathbf{x}\mathbf{x}'})$
  until CONVERGENCE$(\mathbf{v}_{k+1}, \mathbf{v}_k)$
  $\mathbf{v} \leftarrow \mathbf{v}_{k+1}$
  return $\hat{v}^*(\mathbf{x}) = l^0(\mathbf{x})dt - \log(\exp(e^{-\gamma dt}\mathbf{v}^\top) \cdot \mathbf{W} \cdot \mathbf{k}_{\mathbf{x}}(\mathbf{x}))$

$O(n^3)$, and each recurrence is now $O(n^3)$. Moreover, we can now express our algorithm more compactly in terms of matrix multiplications and linear algebra. These two features have the practical consequence of greatly improving computational efficiency.

Given an estimation of the optimal cost-to-go, we compute the estimated optimal policy using the following formula:

$$\hat{\pi}^* = -\Sigma^0 \mathbf{B}^\top \nabla_{\mathbf{x}} \hat{v}^*. \tag{5.10}$$

Computing the optimal policy using this formula necessitates the consideration of two additional issues. Firstly, this formula is dependent on knowing the true parameters Σ^0 and $\mathbf{B}$. On one hand, $\mathbf{B}$ is indeed a necessary model parameter. On the other hand, Σ^0 can be estimated, and we have found in experiments that when Σ^0 is assumed to be state independent and therefore that the conditional embedding estimates may be averaged, good estimates $\hat{\Sigma}^0$ can be found. That being said, for the simplicity of analysis and presentation, in the rest of this thesis we will assume that Σ^0 is known.

Secondly, we now require that $\hat{v}^*$ be differentiable. Consider again equation 5.9, where we now drop the indices of the recurrence relation, reexpressing the relation as a differential Bellman equation. If our state dependent loss rate l^0 is differentiable, then $\hat{v}^*$ is differentiable if and only if the second term is differentiable. Thankfully, this term is simply a differentiable function of the kernel, and so will indeed be differentiable if the underlying kernel is differentiable. If we define $\beta^\top = \exp(e^{-\gamma dt}\mathbf{v}^\top) \cdot \mathbf{W}$, we can reexpress the gradient of

the optimal cost-to-go as

$$\begin{aligned}\nabla_{\mathbf{x}}\hat{v}^*(\mathbf{x}) &= \nabla_{\mathbf{x}} l^0(\mathbf{x})dt - \nabla_{\mathbf{x}}\log(\beta^\top \cdot \mathbf{k}_{\mathbf{x}}(\mathbf{x})) \\ &= \nabla_{\mathbf{x}} l^0(\mathbf{x})dt - \nabla_{\mathbf{x}}\log(\sum_{i=1}^{n}\beta_i k(\mathbf{x}_i,\mathbf{x})) \\ &= \nabla_{\mathbf{x}} l^0(\mathbf{x})dt - \frac{\sum_{i=1}^{n}\beta_i \nabla_{\mathbf{x}} k(\mathbf{x}_i,\mathbf{x})}{\sum_{i=1}^{n}\beta_i k(\mathbf{x}_i,\mathbf{x})}. \end{aligned} \tag{5.11}$$

Our kernel of choice, the Gaussian, is an example of a differentiable kernel. We can express its gradient as follows:

$$\begin{aligned}\frac{\partial k(\mathbf{x},\mathbf{y})}{\partial x_i} &= \frac{\partial e^{-\frac{1}{2}\left(\frac{\|\mathbf{x}-\mathbf{y}\|}{a}\right)^2}}{\partial x_i} \\ &= -\frac{1}{2w^2}k(\mathbf{x},\mathbf{y})\frac{\sum_{j=1}^{n}\partial(x_j-y_j)^2}{\partial x_i} \\ &= \frac{(y_i-x_i)}{w^2}k(\mathbf{x},\mathbf{y}). \end{aligned} \tag{5.12}$$

Therefore, by using algorithm 3 to compute $\hat{v}^*$, we can apply equations 5.11 and 5.12 to computing $\nabla_{\mathbf{x}}\hat{v}^*$. Then given $\mathbf{B}$ and Σ^0, we can compute $\hat{\pi}^*$ using equation 5.10. If one does not have an analytic expression for $\nabla_{\mathbf{x}}k$, one can naturally still rely on finite difference methods to approximate $\nabla_{\mathbf{x}}\hat{v}^*$.

5.2 Kernelized Ergodic Control

In infinite horizon average cost problems, we bound the optimal cost-to-go by attempting to minimize the average, rather than total, loss expected over infinite futures. For a controlled Ito diffusion, the reformulated objective is

$$\lim_{\tau\to\infty}\frac{1}{\tau}E^{\pi}\left(\int_0^{\tau} l(\mathbf{x}_t,\pi(\mathbf{x}_t))dt \mid \mathbf{x}_0 = x\right), \tag{5.13}$$

resulting in the following average cost HJB:

$$a = \min_{\mathbf{u}}[l(\mathbf{u}) + \mu^\top(\mathbf{u})\nabla_{\mathbf{x}}v^* + \frac{1}{2}\mathrm{Tr}(\sigma^2(\mathbf{u})\nabla_{\mathbf{xx}}v^*)], \tag{5.14}$$

where a is the average expected loss, and v^* now represents a differential cost-to-go, where the 'true' cost-to-go is $a + v^*$.

By considering the form of the objective, one can see that solutions of the average cost HJB are determined only by the costs of some long term, equilibrium behaviour. This notion of long term equilibrium behaviour is formally studied by the mathematics of ergodic theory, and for this reason, the problem of minimizing objective 5.13 is often referred to as *ergodic control*.

Ergodic control has advantages and disadvantages in comparison with discounted cost control. On one hand, ergodic control algorithms do not require the parameter γ. Moreover, algorithms for solving the ergodic HJB return the average expected cost a, which is a powerful measure of the performance of a solution.

On the other hand, the ergodic objective does not penalize transient behaviour. As such, solutions which require large amounts of time in order to achieve the desired equilibrium state will not be penalized relative to ones which achieve the same state in a smaller time frame. In discounted cost problems, γ adds a degree of urgency to the cost-to-go.

Practically speaking, discounted cost control has its largest advantage in value iteration. The discounted cost HJB can be approximated by this simple and well understood algorithm, whereas a standard algorithm for average cost value iteration has yet to appear. Perhaps surprisingly, in attempting to formalize the computation of the linear Bellman equations of section 3.3, one finds that while the discounted cost approach leads to a less theoretically robust algorithm, ergodic control solutions become strikingly simple to compute [15].

The linearized form of the ergodic HJB is

$$a = l^0 - \frac{1}{z}\left((\mu^0)^\top \nabla_{\mathbf{x}} z + \frac{1}{2}\operatorname{Tr}(\Sigma \nabla_{\mathbf{xx}} z)\right), \tag{5.15}$$

which, by theorem 3.3 we can approximate with the associated linear differential Bellman equation

$$e^{-a} z = \exp(-l^0 dt) E^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(z). \tag{5.16}$$

and which, by modelling the conditional expectation using RKHS embeddings, we can estimate as

$$e^{-\hat{a}} \hat{z} = \exp(-l^0 dt) \hat{E}^0_{\mathbf{x}_{t+dt} \mid \mathbf{x}_t}(\hat{z}), \tag{5.17}$$

Given a sample from the passive dynamics $\mathscr{S}^0_{dt}$, we may express the sample based estimate of equation 5.17 as

$$e^{-\hat{a}} \mathbf{z} = \mathbf{R} \cdot \mathbf{W} \cdot \mathbf{K}_{\mathbf{xx}'} \cdot \mathbf{z}, \tag{5.18}$$

where $\mathbf{z}_{(i)} = z(\mathbf{x}'_i)$, $\mathbf{R}_{(i,i)} = \exp(-l^0(\mathbf{x}'_i) dt)$, $\mathbf{K}_{\mathbf{xx}',(i,j)} = k(\mathbf{x}_i, \mathbf{x}'_j)$, and $\mathbf{W}$ is as defined in theorem 4.2. In this way we have reduced the computation of the

Algorithm 4 Linearized Ergodic Control

Require: $l^0, \mathscr{S}^0_{dt}, k_{\mathbf{x}}, \lambda$
  $\mathbf{W} \leftarrow (\mathbf{K}_{\mathbf{x}} + \lambda n \mathbf{I})^{-1}$
  $\mathbf{K}_{\mathbf{x}\mathbf{x}'} \leftarrow [\mathbf{k}_{\mathbf{x}}(\mathbf{x}'_1), .., \mathbf{k}_{\mathbf{x}}(\mathbf{x}'_n)]$
  $\mathbf{R} \leftarrow \text{Diag}([l^0(\mathbf{x}'_1), .., l^0(\mathbf{x}'_n)])$
  $\mathbf{z}_0 \leftarrow \mathbf{1}$
  repeat
    $\mathbf{z}_{k+1} \leftarrow \frac{\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}\mathbf{z}_k}{\|\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}\mathbf{z}_k\|}$
  until CONVERGENCE$(\mathbf{z}_{k+1}, \mathbf{z}_k)$
  $\mathbf{z} \leftarrow \mathbf{z}_{k+1}$
  return $e^{-\hat{a}} = \|\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}\mathbf{z}\|, \hat{z}(\mathbf{x}) = \exp(-l^0(\mathbf{x})dt)\cdot\mathbf{z}\cdot\mathbf{W}\cdot\mathbf{k}_{\mathbf{x}}(\mathbf{x})$

solution of the linear HJB to the computation of the eigenvector $\mathbf{z}$ of $\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}$ with eigenvalue e^{-a}. As discussed in [15] and [1], a simple and appropriate method for solving this equation is that of *power iteration*.

Power iteration returns the largest eigenvalue and associated normalized eigenvector in the given linear equation. Applied to our particular problem, power iteration involves iterating the following recurrence relation until convergence:

$$\mathbf{z}_{k+1} = \frac{\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}\cdot\mathbf{z}_k}{\|\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}\cdot\mathbf{z}_k\|} \tag{5.19}$$

The resulting value $\mathbf{z}_\infty$ is our desired eigenvector, and $\|\mathbf{R}\cdot\mathbf{W}\cdot\mathbf{K}_{\mathbf{x}\mathbf{x}'}\cdot\mathbf{z}_\infty\| = e^{-\hat{a}}$ is its eigenvalue. The eigenvalue in turn is an invertible function of $\hat{a}$, thus allowing us to recover the average loss. We summarize this approach in algorithm 4.

Finally, we may express our estimated optimal policy as

$$\hat{\pi}^* = \Sigma^0\mathbf{B}^\top\frac{\nabla_{\mathbf{x}}\hat{z}}{\hat{z}}. \tag{5.20}$$

For computing the gradient of the desirability we may then rely on the same methods as presented in the previous section.

6 Simulations

In this chapter we will test the three algorithms developed in chapter 5 against a simple underpowered pendulum task. Beginning at rest at the stable fixed point, the goal in this task is to swing up and hold the pendulum at its unstable fixed point. In order to make the task challenging, the actuator of the pendulum is designed to be underpowered, such that momentum must first be accumulated before the unstable fixed point can be reached.

In all cases we will use a loss rate of the form $l(\theta, \dot{\theta}, u) = l^0(\theta, \dot{\theta}) + \frac{1}{2}u^2(\Sigma^0)^{-1}$. Where $t(\theta) = \frac{1}{2}ml^2\dot{\theta}^2$ is the kinetic and $u(\theta) = -mgl\cos(\theta)$ the potential energy of the system, we define the state dependent loss rate as $l^0(\theta, \dot{\theta}) = t(\dot{\theta}) - u(\theta)$. Minimizing this loss rate results in $\dot{\theta} \to 0$ and $\theta \to \pi$, and therefore this loss rate drives the pendulum to reach as high as possible and stabilize its position in order to minimize the accumulated loss.

We begin with algorithm 2, as demonstrated in figure 6.1. The differential value approach, as originally developed in [3], solves the control problem as desired. We visualize three ways in which the algorithm solves the problem, and include simulation statistics as well as various model parameters in the text box.

The plots are organized as follows: At the top left are the statistics and parameters. At the top right are the energies over time of the simulation. At the bottom left are the vector fields of the passive drift in grey, and the controlled drift resulting from the estimated policy $\hat{\pi}$ in black. At the bottom right are contour lines of the estimated cost-to-go $\hat{v}^*$. In both the bottom plots a simulation from the controlled drift is displayed in black, with initial condition indicated by the empty circle, and the state of minimal cost-to-go indicated by the blue circle.

The calculated statistics yield one feature worth discussing. The error rates for the underlying model in this simulation are significantly higher than what was found in the simulation of figure 4.4. The primary reason for this is that in figure 4.4 we avoided sampling near the transition between $-\pi$ and π in order to yield results which could be appropriately visualized. Here, we sample around this region, resulting in the increase in error. Nevertheless, with our parameters somewhat increased, the model performs sufficiently well.

In the energy plot we see that the policy manages adds energy to the system, driving it to the point of maximum potential, at which time it stabilizes the fixed point. Inspection of the drifts depicted by the vector fields shows how given possible actions $\mathcal{U} = \{-3.5J, 3.5J\}$, the policy decides whether to

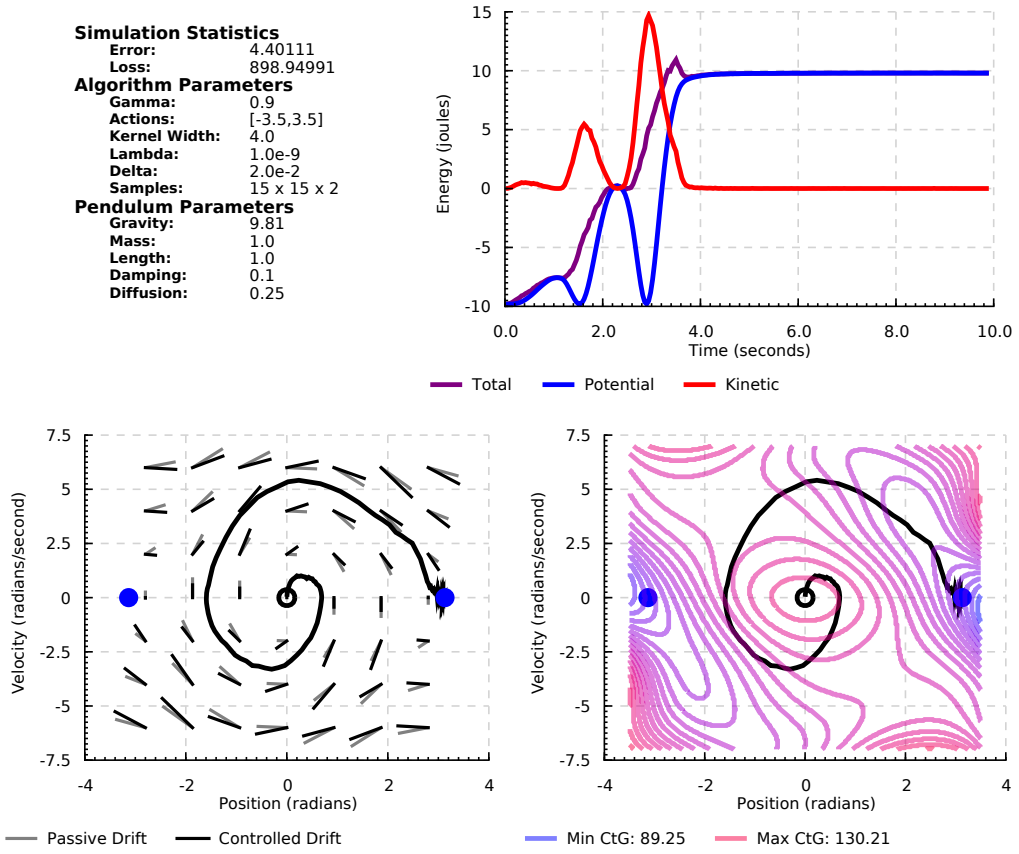

Figure 6.1: **Differential Value Iteration Simulation**

increase or decrease the velocity at any point.

Finally, the contour plot of the optimal cost-to-go yields the clearest display of what the algorithm has learned. Given the limited set of actions $\mathscr{U}$, the optimal cost-to-go is designed to guide a pendulum following the optimal policy to the place of minimal loss. The simulation of such a controlled pendulum is displayed over top of the contour lines of the optimal cost-to-go, one can see how the pendulum does indeed follow the gradient to $\mathbf{x} = [\theta, \dot{\theta}] = [\pi, 0]$. Visual inspection of the optimal cost-to-go reveals a policy which seems capable of guiding the pendulum to the goal from a variety of initial conditions.

Unfortunately, our efforts with the other two algorithms were not successful. Although the algorithms would often convergence and produce somewhat meaningful results, and moreover that the convergence of these algorithms has indeed proven extremely fast as shown, these results are highly unstable and are not suitable for presentation.

The instability of the results is caused by a number of subtle numerical is-

sues. Since the cost-to-go can take on extreme values, the exponentiation and logarithm operations lead to a great deal of numerical instability unless the parameters are finely tuned. Moreover, the formula for computing conditional RKHS embeddings can lead to negative weights. This in turn may result in problematic estimates such as the estimated expectation of a positive function (i.e. exponentiation) resulting in a negative value. There are more subtle numeric issues as well which we so far have been unable to disentangle.

There are nevertheless many avenues to explore in attempting to resolve these issues. In many papers on path integral control, e.g. [8, 12], a parameter λ is provided for rescaling the cost of control. However, the way in which this lambda appears in the relevant equations breaks the correspondence with the KL control equations, and as such, we would not be able to apply this approach directly in our formulation. Neveretheless, other recent work has developed some of the necessary formulas for controlling the magnitude of the cost functions [1].

Finally, in order to guarantee the positivity of the conditional RKHS embeddings, various tricks can be employed. Simply setting negative values to 0 and rescaling, as suggested in [3], does appear to improve performance. Nevertheless, the resulting algorithms remained unable to solve our basic control problems. Another approach would be to use some form of principled matrix approximation, such as suggested in [4], with appropriate provisions made in order to guarantee positive values. We hope to test these various approaches in the future.

7 Conclusion

In this thesis we explored recent developments in efficient stochastic optimal control. We began by developing the basic physics for describing stochastic robot control problems. We then developed the theoretical tools to model continuous space, continuous time, stochastic control problems. In order to compute the conditional expectations which were critical to our control equations, we presented recent work in kernel methods, and applied them to a vector field modelling task. We then developed a series of computable algorithms based on the theory of the preceding sections, and finished with our simulations.

Although we were only able to produce a successfully policy in the non KL control case, the initial theoretical work done in developing our KL control based algorithms is sound. Both the application of value iteration to the KL control problem, and the formation of an eigenvector problem out of the matrices driving the kernel methods, are theoretically justified approaches which require more rigorous analysis in order to solve lingering technical problems.

Solving this scaling and negativity problems would do much towards granting the KL control algorithms the numerical stability required in order to be easily be applied to a variety of systems. Once these algorithms have been demonstrated as stable, there would still remain many other other approaches worth exploring to improve performance, such as sparsification and alternative kernels. Such results could form the basis of applying the methods of this thesis to ever more complicated problems, allowing us to greatly push the boundaries of the curse of dimensionality.

8 Bibliography

[1] Bierkens, J. and B. Kappen (2011). Kl-learning: Online solution of kullback-leibler control problems. *arXiv preprint arXiv:1112.1996*.

[2] Fukumizu, K., L. Song, and A. Gretton (2010, September). Kernel bayes' rule. *arXiv:1009.5736*.

[3] Grunewalder, S., G. Lever, L. Baldassarre, M. Pontil, and A. Gretton (2012). Modelling transition dynamics in mdps with rkhs embeddings. *arXiv preprint arXiv:1206.4655*.

[4] Grünewälder, S., G. Lever, L. Baldassarre, S. Patterson, A. Gretton, and M. Pontil (2012). Conditional mean embeddings as regressors. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, Volume 2, pp. 1823–1830.

[5] Kappen, B. (2007). An introduction to stochastic control theory, path integrals and reinforcement learning.

[6] Kappen, H. J. (2005). Linear theory for control of nonlinear stochastic systems. *Physical review letters 95*(20), 200201.

[7] Kappen, H. J., V. Gómez, and M. Opper (2012, February). Optimal control as a graphical model inference problem. *Machine Learning 87*(2), 159–182.

[8] Rawlik, K., M. Toussaint, and S. Vijayakumar (2012). Path integral control by reproducing kernel hilbert space embedding. *Arxiv preprint arXiv:1208.2523*.

[9] Song, L., J. Huang, A. Smola, and K. Fukumizu (2009). Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 961–968.

[10] Szepesvári, C. (2010). Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning 4*(1), 1–103.

[11] Tedrake, R. (2009). Underactuated robotics: Learning, planning, and control for efficient and agile machines course notes for MIT 6.832. *Working draft edition*.

[12] Theodorou, E., J. Buchli, and S. Schaal (2010a). A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research 9999*, 3137–3181.

[13] Theodorou, E., J. Buchli, and S. Schaal (2010b). Reinforcement learning of motor skills in high dimensions: a path integral approach. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 2397–2403.

[14] Todorov, E. (2006). Optimal control theory. *Bayesian brain: probabilistic approaches to neural coding*, 269–298.

[15] Todorov, E. (2009). Efficient computation of optimal actions. *Proceedings of the national academy of sciences 106*(28), 11478–11483.

[16] Zhang, L., W. Zhou, and L. Jiao (2004). Wavelet support vector machine. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 34*(1), 34–39.