An introduction to stochastic control theory, path integrals and reinforcement learning

Hilbert J. Kappen

Department of Biophysics, Radboud University, Geert Grooteplein 21, 6525 EZ Nijmegen

Abstract. Control theory is a mathematical description of how to act optimally to gain future rewards. In this paper I give an introduction to deterministic and stochastic control theory and I give an overview of the possible application of control theory to the modeling of animal behavior and learning. I discuss a class of non-linear stochastic control problems that can be efficiently solved using a path integral or by MC sampling. In this control formalism the central concept of cost-to-go becomes a free energy and methods and concepts from statistical physics can be readily applied.

Keywords: Stochastic optimal control, path integral control, reinforcement learning
PACS: 05.45.-a 02.50.-r 45.80.+r

INTRODUCTION

Animals are well equipped to survive in their natural environments. At birth, they already possess a large number of skills, such as breathing, digestion of food and elementary processing of sensory information and motor actions.

In addition, they acquire complex skills through learning. Examples are the recognition of complex constellations of sensory patterns that may signal danger or food or pleasure, and the execution of complex sequences of motor commands, whether to reach for a cup, to climb a tree in search of food or to play the piano. The learning process is implemented at the neural level, through the adaptation of synaptic connections between neurons and possibly other processes.

It is not well understood how billions of synapses are adapted without central control to achieve the learning. It is known, that synaptic adaptation results from the activity of the nearby neurons, in particular the pre- and post-synaptic neuron that it connects. The adaptation is quite complex: it depends on the temporal history of the neural activity; it is different for different types of synapses and brain areas and the outcome of the learning is not a static synaptic weight, but rather an optimized dynamical process that implements a particular transfer between the neurons [1, 2].

The neural activity that causes the synaptic adaptation is determined by the sensory data that the animal receives and by the motor commands that it executes. These data are in turn determined by the behavior of the animal itself, i.e. which objects it looks at and which muscles it contracts. Thus, learning affects behavior and behavior affects learning.

In this feedback circuit, the learning algorithms itself is still to be specified. The learning algorithm will determine what adaptation will take place given a recent history of neural activity. It is most likely that this algorithm is determined genetically. Our genes are our record of our successes and failures throughout evolution. If you have good

genes you will learn better and therefore have a better chance at survival and the creation of off-spring. The genetic information may not only affect the learning algorithm, but also affects our 'innate' tendency to choose the environment that we live in. For instance, a curious animal will tend to explore richer and more challenging environments and its brain will therefore adapt to a more complex and more varied data set, increasing the level of the skills that the animal learns.

Such genetic influences have been also observed in humans. For instance, it has been observed that the heritability of intelligence increases with age: as we grow older, our intelligence (in the sense of reasoning and novel problem-solving ability) reflects our genotype more closely. This could be explained by the fact that as our genes determine our ability to learn and our curiosity to explore novel, diverse, environments, such learning will make us smarter the older we grow [3, 4]. On the other hand, if learning would not have a genetic component, and would only be determined by the environment, one would predict the opposite: the influence of the environment on our intelligence increases with age, and therefore decreases the relative influence of our genetic material with which we are born.

The most influential biological learning paradigm is Hebbian learning [5]. This learning rule was originally proposed by the psychologist Hebb to account for the learning behavior that is observed in learning experiments with animals and humans and that can account for simple cognitive behaviors such as habituation and classical conditioning [1]. Hebbian learning states that neurons increase the synaptic connection strength between them when they are both active at the same time and slowly decrease the synaptic strength otherwise. The rationale is that when a presynaptic spike (or the stimulus) contributes to the firing of the post synaptic neuron (the response), it is likely that its contribution is of some functional importance to the animal and therefore the synapse should be strengthened. If not, the synapse is probably not very important and its strength is decreased. The mechanism of Hebbian learning has been confirmed at the neural level in some cases [6], but is too simple as a theory of synaptic plasticity in general. In particular, synapses display an interesting history dependent dynamics with characteristic time scales of several msec to hours. Hebbian learning is manifest in many areas of the brain and most neural network models use the Hebb rule in a more or less modified way to explain for instance the receptive fields properties of sensory neurons in visual and auditory cortical areas or the formation cortical maps (see [7] for examples).

Hebbian learning is instantaneous in the sense that the adaptation at time t is a function of the neural activity or the stimuli at or around time t only. This is sufficient for learning time-independent mappings such as receptive fields, where the correct response at time t only depends on the stimulus at or before time t. The Hebbian learning rule can be interpreted as a way to achieve this optimal instantaneous stimulus response behavior.

---

[1] Habituation is the phenomenon that an animal gets accustomed to a new stimulus. For instance, when ringing a bell, a dog will turn its head. When repeated many times, the dog will ignore the bell and no longer turn its head. Classical conditioning is the phenomenon that a stimulus that does not produce a response can be made to produce a response if it has been co-presented with another stimulus that does produce a response. For instance, a dog will not salivate when hearing a bell, but will do so when seeing a piece of meat. When the bell and the meat are presented simultaneously during a repeated number of trials, afterwards the dog will also salivate when only the bell is rung.

However, many tasks are more complex than simple stimulus-response behavior. They require a sequence of responses or actions and the success of the sequence is only known at some future time. Typical examples are any type of planning task such as the execution of a motor program or searching for food.

Optimizing a sequence of actions to attain some future goal is the general topic of control theory [8, 9]. It views an animal as an automaton that seeks to maximize expected reward (or minimize cost) over some future time period. Two typical examples that illustrate this are motor control and foraging for food. As an example of a motor control task, consider throwing a spear to kill an animal. Throwing a spear requires the execution of a motor program that is such that at the moment that the spear releases the hand, it has the correct speed and direction such that it will hit the desired target. A motor program is a sequence of actions, and this sequence can be assigned a cost that consists generally of two terms: a path cost, that specifies the energy consumption to contract the muscles in order to execute the motor program; and an end cost, that specifies whether the spear will kill the animal, just hurt it, or misses it altogether. The optimal control solution is a sequence of motor commands that results in killing the animal by throwing the spear with minimal physical effort. If x denotes the state space (the positions and velocities of the muscles), the optimal control solution is a function $u(x,t)$ that depends both on the actual state of the system at each time and also depends explicitly on time.

When an animal forages for food, it explores the environment with the objective to find as much food as possible in a short time window. At each time t, the animal considers the food it expects to encounter in the period $[t,t+T]$. Unlike the motor control example, the time horizon recedes into the future with the current time and the cost consists now only of a path contribution and no end-cost. Therefore, at each time the animal faces the same task, but possibly from a different location in the environment. The optimal control solution $u(x)$ is now time-independent and specifies for each location in the environment x the direction u in which the animal should move.

Motor control and foraging are examples of finite horizon control problems. Other reward functions that are found in the literature are infinite horizon control problems, of which two versions exist. One can consider discounted reward problems where the reward is of the form $C = \langle \sum_{t=0}^{\infty} \gamma^t R_t \rangle$ with $0 < \gamma < 1$. That is, future rewards count less than immediate rewards. This type of control problem is also called reinforcement learning (RL) and is popular in the context of biological modeling. Reinforcement learning can be applied even when the environment is largely unknown and well-known algorithms are temporal difference learning [10], Q-learning [11] and the actor-critic architecture [12]. RL has also been applied to engineering and AI problems, such as an elevator dispatching task [13], robotic jugglers [14] and to play back-gammon [15]. One can also consider infinite horizon average rewards $C = \lim_{h\to\infty} \frac{1}{h} \left\langle \sum_{t=0}^{h} R_t \right\rangle$. A disadvantage of this cost is that the optimal solution is insensitive to short-term gains since it makes a negligible contribution to the infinite average. Both these infinite horizon control problems have time-independent optimal control solutions.

Note, that the control problem is naturally stochastic in nature. The animal does not typically know where to find the food and has at best a probabilistic model of the expected outcomes of its actions. In the motor control example, there is noise in the relation between the muscle contraction and the actual displacement of the joints. Also, the environment changes over time which is a further source of uncertainty. Therefore,

the best the animal can do is to compute the optimal control sequence with respect to the expected cost. Once, this solution is found, the animal executes the first step of this control sequence and re-estimates its state using his sensor readings. In the new state, the animal recomputes the optimal control sequence using the expected cost, etc.

There is recent work that attempts to link control theory, and in particular RL, to computational strategies that underly decision making in animals [16, 17]. This novel field is sometimes called neuro-economics: to understand the mechanisms of decision making at the cellular and circuit level in the brain. Physiological studies locate these functions across both frontal and parietal cortices. Typically, tasks are studied where the behavior of the animal depends on reward that is delayed in time. For instance, dopamine neurons respond to reward at the time that the reward is given. When on repeated trials the upcoming reward is 'announced' by a conditioning stimulus (CS), the dopamine neurons learn to respond to the CS as well (see for instance [18] for a review). This type of conditioning is adaptive and depends on the timing of the CS relative to the reward and the amount of information the CS contains about the reward. The neural representation of reward, preceding the actual occurrence of the reward confirms the notion that some type of control computation is being performed by the brain.

In delayed reward tasks, one thus finds that learning is based on reward signals, also called value signals, and one refers to this type of learning as value-based learning, to be distinguished from the traditional Hebbian perception-based learning. In perception-based learning, the learning is simply Hebbian and reinforces correlations between the stimulus and the response, action or reward at the same time. In value-based learning, a value representation is first built from past experiences that predicts the future reward of current actions (see [16] for a review).

Path integral control

The general stochastic control problem is intractable to solve and requires an exponential amount of memory and computation time. The reason is that the state space needs to be discretized and thus becomes exponentially large in the number of dimensions. Computing the expectation values means that all states need to be visited and requires the summation of exponentially large sums. The same intractabilities are encountered in reinforcement learning. The most efficient RL algorithms (TD(λ) [19] and Q learning [11]) require millions of iterations to learn a task.

There are some stochastic control problems that can be solved efficiently. When the system dynamics is linear and the cost is quadratic (LQ control), the solution is given in terms of a number of coupled ordinary differential (Ricatti) equations that can be solved efficiently [8]. LQ control is useful to maintain a system such as for instance a chemical plant, operated around a desired point in state space and is therefore widely applied in engineering. However, it is a linear theory and too restricted to model the complexities of animal behavior. Another interesting case that can be solved efficiently is continuous control in the absence of noise [8]. One can apply the so-called Pontryagin Maximum Principle [20], which is a variational principle, that leads to a coupled system of ordinary differential equations with boundary conditions at both initial and final time.

Although this deterministic control problem is not intractable in the above sense, solving the differential equation can still be rather complex in practice.

Recently, we have discovered a class of continuous non-linear stochastic control problems that can be solved more efficiently than the general case [21, 22]. These are control problems with a finite time horizon, where the control acts linearly and additive on the dynamics and the cost of the control is quadratic. Otherwise, the path cost and end cost and the intrinsic dynamics of the system are arbitrary. These control problems can have both time-dependent and time-independent solutions of the type that we encountered in the examples above. The control problem essentially reduces to the computation of a path integral, which can be interpreted as a free energy. Because of its typical statistical mechanics form, one can consider various ways to approximate this path integral, such as the Laplace approximation [22], Monte Carlo sampling [22], mean field approximations or belief propagation [23]. Such approximate computations are sufficiently fast to be possibly implemented in the brain.

Also, one can extend this control formalism to multiple agents that jointly solve a task. In this case the agents need to coordinate their actions not only through time, but also among each other. It was recently shown that the problem can be mapped on a graphical model inference problem and can be solved using the junction tree algorithm. Exact control solutions can be computed for instance with hundreds of agents, depending on the complexity of the cost function [24, 23].

Non-linear stochastic control problems display features not shared by deterministic control problems nor by linear stochastic control. In deterministic control, only one globally optimal solution exists. In stochastic control, the optimal solution is a weighted mixture of suboptimal solutions. The weighting depends in a non-trivial way on the features of the problem, such as the noise and the horizon time and on the cost of each solution. This multi-modality leads to surprising behavior is stochastic optimal control. For instance, the phenomenon of obstacle avoidance for autonomous systems not only needs to make the choice of whether to turn left or right, but also *when* such decision should be made. When the obstacle is still far away, no action is required, but there is a minimal distance to the obstacle when a decision should be made. This example was treated in [21] and it was shown that the decision is implemented by spontaneous symmetry breaking where one solution (go straight ahead) breaks in two solutions (turn left or right).

Exploration

Computing optimal behavior for an animal consists of two difficult subproblems. One is to compute the optimal behavior for a given environment, assuming that the environment is known to the animal. The second problem is to learn the environment. Here, we will mainly focus on the first problem, which is typically intractable and where the path integral approach can give efficient approximate solutions. The second problem is complicated by the fact that not all of the environment is of interest to the animal: only those parts that have high reward need to be learned. It is intuitively clear that a suboptimal control behavior that is computed by the animal, based on the limited part

of the environment that he has explored, may be helpful to select the more interesting parts of the environment. But clearly, part of the animals behavior should also be purely explorative with the hope to find even more rewarding parts of the environment. This is known as the exploration-exploitation dilemma.

Here is an example. Suppose that you are reasonably happy with your job. Does it make sense to look for a better job? It depends. There is a certain amount of agony associated with looking for a job, getting hired, getting used to the new work and moving to another city. On the other hand, if you are still young and have a life ahead of you, it may well be worth the effort. The essential complication here is that the environment is not known and that on the way from the your current solution to the possibly better solution one may have to accept a transitionary period with relative high cost.

If the environment is known, there is no exploration issue and the optimal strategy can be computed, although this will typically require exponential time and/or memory. As we will see in the numerical examples at the end of this paper, the choice to make the transition to move to a better position is optimal when you have a long life ahead, but it is better to stay in your current position if you have not much time left. If the environment is not known, one should explore 'in some way' in order to learn the environment. The optimal way to explore is in general not part of the control problem.

Outline

In this review, we aim to give a pedagogical introduction to control theory. For simplicity, we will first consider the case of discrete time and discuss the dynamic programming. Subsequently, we consider continuous time control problems. In the absence of noise, the optimal control problem can be solved in two ways: using the Pontryagin Minimum Principle (PMP) [20] which is a pair of ordinary differential equations that are similar to the Hamilton equations of motion or the Hamilton-Jacobi-Bellman (HJB) equation which is a partial differential equation [25].

In the presence of Wiener noise, the PMP formalism has no obvious generalization (see however [26]). In contrast, the inclusion of noise in the HJB framework is mathematically quite straight-forward. However, the numerical solution of either the deterministic or stochastic HJB equation is in general difficult due to the curse of dimensionality.

Subsequently, we discuss the special class of control problems introduced in [21, 22]. For this class of problems, the non-linear Hamilton-Jacobi-Bellman equation can be transformed into a linear equation by a log transformation of the cost-to-go. The transformation stems back to the early days of quantum mechanics and was first used by Schrödinger to relate the Hamilton-Jacobi formalism to the Schrödinger equation. The log transform was first used in the context of control theory by [27] (see also [9]).

Due to the linear description, the usual backward integration in time of the HJB equation can be replaced by computing expectation values under a forward diffusion process. The computation of the expectation value requires a stochastic integration over trajectories that can be described by a path integral. This is an integral over all trajectories starting at x,t, weighted by $\exp(-S/\nu)$, where S is the cost of the path (also know as the Action) and ν is the size of the noise.

The path integral formulation is well-known in statistical physics and quantum mechanics, and several methods exist to compute path integrals approximately. The Laplace approximation approximates the integral by the path of minimal S. This approximation is exact in the limit of $\nu \to 0$, and the deterministic control law is recovered.

In general, the Laplace approximation may not be sufficiently accurate. A very generic and powerful alternative is Monte Carlo (MC) sampling. The theory naturally suggests a naive sampling procedure, but is also possible to devise more efficient samplers, such as importance sampling.

We illustrate the control method on two tasks: a temporal decision task, where the agent must choose between two targets at some future time; and a receding horizon control task. The decision task illustrates the issue of spontaneous symmetry breaking and how optimal behavior is qualitatively different for high and low noise. The receding horizon problem is to optimize the expected cost over a fixed future time horizon. This problem is similar to the RL discounted reward cost. We have therefore also included a section that introduces the main ideas of RL.

We start by discussing the most simple control case, which is the finite horizon discrete time deterministic control problem. In this case the optimal control explicitly depends on time. The derivations in this section are based on [28]. Subsequently, we discuss deterministic, stochastic continuous time control and reinforcement learning. Finally, we give a number of illustrative numerical examples.

DISCRETE TIME CONTROL

Consider the control of a discrete time dynamical system:

$$x_{t+1} = f(t, x_t, u_t), \quad t = 0, 1, \ldots, T \tag{1}$$

x_t is an n-dimensional vector describing the *state* of the system and u_t is an m-dimensional vector that specifies the *control* or *action* at time t. Note, that Eq. 1 describes a noiseless dynamics. If we specify x at $t = 0$ as x_0 and we specify a sequence of controls $u_{0:T} = u_0, u_1, \ldots, u_T$, we can compute future states of the system $x_1, \ldots, x_{T+1}$ recursively from Eq.1.

Define a cost function that assigns a cost to each sequence of controls:

$$C(x_0, u_{0:T}) = \sum_{t=0}^{T} R(t, x_t, u_t) \tag{2}$$

$R(t, x, u)$ can be interpreted as a deterministic cost that is associated with taking action u at time t in state x or with the expected cost, given some probability model (as we will see below). The problem of optimal control is to find the sequence $u_{0:T}$ that minimizes $C(x_0, u_{0:T})$.

The problem has a standard solution, which is known as dynamic programming. Introduce the *optimal cost to go*:

$$J(t, x_t) = \min_{u_{t:T}} \sum_{s=t}^{T} R(s, x_s, u_s) \tag{3}$$

which solves the optimal control problem from an intermediate time t until the fixed end time T, starting at an arbitrary location x_t. The minimum of Eq. 2 is given by $J(0,x_0)$.

One can recursively compute $J(t,x)$ from $J(t+1,x)$ for all x in the following way:

$$\begin{aligned} J(T+1,x) &= 0 \qquad (4) \\ J(t,x_t) &= \min_{u_{t:T}} \sum_{s=t}^{T} R(s,x_s,u_s) \\ &= \min_{u_t} \left(R(t,x_t,u_t) + \min_{u_{t+1:T}} \sum_{s=t+1}^{T} R(s,x_s,u_s) \right) \\ &= \min_{u_t} \left(R(t,x_t,u_t) + J(t+1,x_{t+1}) \right) \\ &= \min_{u_t} \left(R(t,x_t,u_t) + J(t+1,f(t,x_t,u_t)) \right) \qquad (5) \end{aligned}$$

The algorithm to compute the optimal control $u^*_{0:T}$, the optimal trajectory $x^*_{1:T}$ and the optimal cost is given by

1. Initialization: $J(T+1,x) = 0$
2. For $t = T,\ldots,0$ and for all x compute

$$\begin{aligned} u^*_t(x) &= \arg\min_u \{ R(t,x,u) + J(t+1,f(t,x,u)) \} \qquad (6) \\ J(t,x) &= R(t,x,u^*_t) + J(t+1,f(t,x,u^*_t)) \qquad (7) \end{aligned}$$

3. For $t = 0,\ldots,T-1$ compute forwards ($x^*_0 = x_0$)

$$x^*_{t+1} = f(t,x^*_t,u^*_t) \qquad (8)$$

Note, that the dynamic programming equations must simultaneously compute $J(t,x)$ for all x. The reason is that $J(t,x)$ is given in terms of $J(t+1,f(t,x,u))$, which is a different value of x. Which x this is, is not known until after the algorithm has computed the optimal control u. The execution of the dynamic programming algorithm is linear in the horizon time T and linear in the size of the state and action spaces.

DETERMINISTIC CONTINUOUS TIME CONTROL

In the absence of noise, the optimal control problem can be solved in two ways: using the Pontryagin Minimum Principle (PMP) [20] which is a pair of ordinary differential equations that are similar to the Hamilton equations of motion or the Hamilton-Jacobi-Bellman (HJB) equation which is a partial differential equation [25]. The latter is very similar to the dynamic programming approach that we have treated before. The HJB approach also allows for a straightforward extension to the noisy case. We will therefore restrict our attention to the HJB description. For further reading see [8, 28].

Consider the control of a dynamical system

$$\dot{x} = f(x,u,t) \qquad (9)$$

The initial state is fixed: $x(t_i) = x_i$ and the final state is free. The problem is to find a control signal $u(t), t_i < t < t_f$, which we denote as $u(t_i \rightarrow t_f)$, such that

$$C(t_i, x_i, u(t_i \rightarrow t_f)) = \phi(x(t_f)) + \int_{t_i}^{t_f} dt R(x(t), u(t), t) \tag{10}$$

is minimal. C consists of an end cost $\phi(x)$ that gives the cost of ending in a configuration x, and a path cost that is an integral over the time trajectories $x(t_i \rightarrow t_f)$ and $u(t_i \rightarrow t_f)$.

We define the *optimal cost-to-go function* from any intermediate time t and state x:

$$J(t,x) \quad = \quad \min_{u(t \rightarrow t_f)} C(t, x, u(t \rightarrow t_f)) \tag{11}$$

For any intermediate time $t', t < t' < t_f$ we get

$$\begin{aligned} J(t,x) &= \min_{u(t \rightarrow t_f)} \left(\phi(x(t_f)) + \int_t^{t'} dt R(x(t), u(t), t) + \int_{t'}^{t_f} dt R(x(t), u(t), t) \right) \\ &= \min_{u(t \rightarrow t')} \left(\int_t^{t'} dt R(x(t), u(t), t) + \min_{u(t' \rightarrow t_f)} \left(\phi(x(t_f)) + \int_{t'}^{t_f} dt R(x(t), u(t), t) \right) \right) \\ &= \min_{u(t \rightarrow t')} \left(\int_t^{t'} dt R(x(t), u(t), t) + J(t', x(t')) \right) \end{aligned}$$

The first line is just the definition of J. In the second line, we split the minimization over two intervals. These are not independent, because the second minimization is conditioned on the starting value $x(t')$, which depends on the outcome of the first minimization. The last line uses again the definition of J.

Setting $t' = t + dt$ with dt infinitesimal small, we can expand $J(t', x(t')) = J(t, x(t)) + \partial_t J(t, x(t)) + \partial_x J(t, x(t)) dx$ and we get

$$J(t,x) \quad = \quad \min_{u(t \rightarrow t+dt)} (R(x, u(t), t) dt + J(t,x) + J_t(t,x) dt + J_x(t,x) f(x, u(t), t) dt)$$

where we have used Eq. 9: $dx = f(x,u,t)dt$. ∂_t and ∂_x denote partial derivatives with respect to t and x, respectively. Note, that the minimization has now reduced over a path of infinitesimal length. In the limit, this minimization over a path reduces to a minimization over a point-wise variable u at time t. Rearranging terms we obtain

$$-J_t(t,x) = \min_u (R(x,u,t) + J_x(t,x) f(x,u,t)) \tag{12}$$

which is the *Hamilton-Jacobi-Bellman Equation*. The equation must be solved with boundary condition for J at the end time: $J(t_f, x) = \phi(x)$, which follows from its definition Eq. 11.

Thus, computing the optimal control requires to solve the partial differential equation 12 for all x backwards in time from t_f to the current time t. The optimal control at the current x, t is given by

$$u(x,t) = \arg\min_u (R(x,u,t) + J_x(t,x) f(x,u,t)) \tag{13}$$

Note, that the HJB approach to optimal control necessarily must compute the optimal control for all values of x at the current time, although in principle the optimal control at the current x value would be sufficient.

Example: Mass on a spring

To illustrate the optimal control principle consider a mass on a spring. The spring is at rest at $z = 0$ and exerts a force proportional to $F = -z$ towards the rest position. Using Newton's Law $F = ma$ with $a = \ddot{z}$ the acceleration and $m = 1$ the mass of the spring, the equation of motion is given by.

$$\ddot{z} = -z + u$$

with u a unspecified control signal with $-1 < u < 1$. We want to solve the control problem: Given initial position and velocity z_i and $\dot{z}_i$ at time t_i, find the control path $u(t_i \rightarrow t_f)$ such that $z(t_f)$ is maximal.

Introduce $x_1 = z, x_2 = \dot{z}$, then

$$\dot{x} = Ax + Bu, \qquad A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

and $x = (x_1, x_2)^T$. The problem is of the above type, with $\phi(x) = C^T x$, $C^T = (-1, 0)$, $R(x,u,t) = 0$ and $f(x,u,t) = Ax + Bu$. Eq. 12 takes the form

$$-J_t = J_x^T Ax - |J_x^T B|$$

We try $J(t,x) = \psi(t)^T x + \alpha(t)$. The HJBE reduces to two ordinary differential equations

$$\begin{aligned} \dot{\psi} &= -A^T \psi \\ \dot{\alpha} &= |\psi^T B| \end{aligned}$$

These equations must be solved for all t, with final boundary conditions $\psi(t_f) = C$ and $\alpha(t_f) = 0$. Note, that the optimal control in Eq. 13 only requires $J_x(x,t)$, which in this case is $\psi(t)$ and thus we do not need to solve α. The solution for ψ is

$$\begin{aligned} \psi_1(t) &= -\cos(t - t_f) \\ \psi_2(t) &= \sin(t - t_f) \end{aligned}$$

for $t_i < t < t_f$. The optimal control is

$$u(x,t) = -\text{sign}(\psi_2(t)) = -\text{sign}(\sin(t - t_f))$$

As an example consider $t_i = 0$, $x_1(t_i) = x_2(t_i) = 0$, $t_f = 2\pi$. Then, the optimal control is

$$\begin{aligned} u &= -1, \quad 0 < t < \pi \\ u &= 1, \quad \pi < t < 2\pi \end{aligned}$$

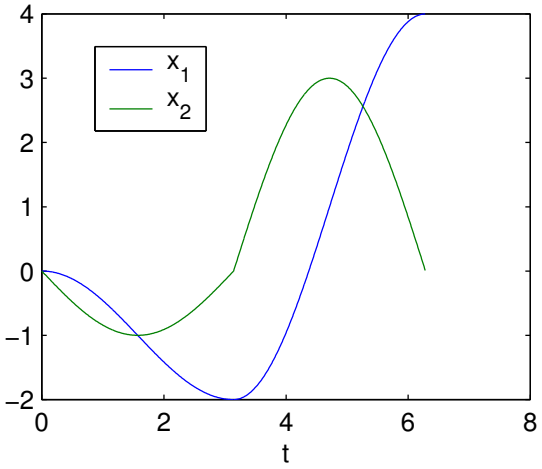

FIGURE 1. Optimal control of mass on a spring such that at $t = 2\pi$ the amplitude is maximal. x_1 is position of the spring, x_2 is velocity of the spring.

The optimal trajectories are for $0 < t < \pi$

$$x_1(t) = \cos(t) - 1, \quad x_2(t) = -\sin(t)$$

and for $\pi < t < 2\pi$

$$x_1(t) = 3\cos(t) + 1, \quad x_2(t) = -3\sin(t)$$

The solution is drawn in fig. 1. We see that in order to excite the spring to its maximal height at t_f, the optimal control is to first push the spring down for $0 < t < \pi$ and then to push the spring up between $\pi < t < 2\pi$, taking maximally advantage of the intrinsic dynamics of the spring.

Note, that since there is no cost associated with the control u and u is hard limited between -1 and 1, the optimal control is always either -1 or 1. This is known as bang-bang control.

STOCHASTIC OPTIMAL CONTROL

In this section, we consider the extension of the continuous control problem to the case that the dynamics is subject to noise and is given by a stochastic differential equation. We restrict ourselves to the one-dimensional example. Extension to n dimensions is straightforward and is treated in [22]. Consider the stochastic differential equation which is a generalization of Eq. 9:

$$dx = f(x(t), u(t), t)dt + d\xi. \tag{14}$$

$d\xi$ is a Wiener processes with $\langle d\xi^2 \rangle = \nu dt$. [2]

Because the dynamics is stochastic, it is no longer the case that when x at time t and the full control path $u(t \to t_f)$ are given, we know the future path $x(t \to t_f)$. Therefore, we cannot minimize Eq. 10, but can only hope to be able to minimize its expectation value over all possible future realizations of the Wiener process:

$$C(x_i, t_i, u(t_i \to t_f)) = \left\langle \phi(x(t_f)) + \int_{t_i}^{t_f} dt R(x(t), u(t), t) \right\rangle_{x_i} \tag{15}$$

The subscript x_i on the expectation value is to remind us that the expectation is over all stochastic trajectories that start in x_i.

The solution of the control problem proceeds as in the deterministic case. One defines the optimal cost-to-go Eq. 11 and obtains as before the recursive relation

$$J(x,t) \quad = \quad \min_{u(t \to t')} \left\langle \int_t^{t'} dt R(x(t), u(t), t) + J(x(t'), t') \right\rangle_x \tag{16}$$

Setting $t' = t + dt$ we can Taylor expand $J(x(t'), t')$ around t, but now to first order in dt and to second order in dx, since $\langle dx^2 \rangle = \mathcal{O}(dt)$. This is the standard Itô calculus argument. Thus,

$$\langle J(x(t+dt), t+dt) \rangle_x \quad = \quad J(x,t) + \partial_t J(x,t) dt + \partial_x J(x,t) f(x,u,t) dt + \frac{1}{2} \partial_x^2 J(x,t) \nu dt$$

Substituting this into Eq. 16 and rearranging terms yields

$$-\partial_t J(x,t) = \min_u \left(R(x,u,t) + f(x,u,t)^T \partial_x J(x,t) + \frac{1}{2} \nu \partial_x^2 J(x,t) \right) \tag{17}$$

which is the *Stochastic Hamilton-Jacobi-Bellman Equation* with boundary condition $J(x, t_f) = \phi(x)$. Eq. 17 reduces to the deterministic HJB equation in the limit $\nu \to 0$.

A linear HJB equation

Consider the special case of Eqs. 14 and 15 where the dynamic is linear in u and the cost is quadratic in u:

$$f(x,u,t) \quad = \quad f(x,t) + u \tag{18}$$

---

[2] A Wiener process can be intuitively understood as the continuum limit of a random walk. Consider ξ on a one-dimensional grid with locations $\xi = 0, \pm d\xi, \pm 2d\xi, \ldots$. Discretize time as $t = 0, \sqrt{dt}, 2\sqrt{dt}, \ldots$. The random walk starts at $\xi = t = 0$ and at each time step moves up or down with displacement $d\xi_i = \pm\sqrt{\nu dt}$. After a large number of N time steps, $\xi = \sum_i d\xi_i$. Since ξ is a sum of a large number of independent contributions, its probability is distributed as a Gaussian. The mean of the distribution $\langle \xi \rangle = 0$, since the mean of each contribution $\langle d\xi_i \rangle = 0$. The variance σ^2 of ξ after N time steps is the sum of the variances: $\sigma^2 = \langle \xi^2 \rangle = \sum_i \langle d\xi_i^2 \rangle = \nu N dt$. The Wiener process is obtained by taking $N \to \infty$ and $dt \to 0$ while keeping the total time $t = N dt$ constant. Instead of choosing $d\xi_i = \pm\sqrt{\nu dt}$ one can equivalently draw $d\xi_i$ from a Gaussian distribution with mean zero and variance νdt.

$$R(x,u,t) \quad = \quad V(x,t) + \frac{R}{2}u^2 \tag{19}$$

with R a positive number. $f(x,t)$ and $V(x,t)$ are arbitrary functions of x and t. In other words, the system to be controlled can be arbitrary complex and subject to arbitrary complex costs. The control instead, is restricted to the simple linear-quadratic form.

The stochastic HJB equation 17 becomes

$$-\partial_t J(x,t) \quad = \quad \min_u \left(\frac{R}{2}u^2 + V(x,t) + (f(x,t)+u)\partial_x J(x,t) + \frac{1}{2}\nu\partial_x^2 J(x,t) \right)$$

Due to the linear-quadratic appearance of u, we can minimize with respect to u explicitly which yields:

$$u = -\frac{1}{R}\partial_x J(x,t) \tag{20}$$

which defines the optimal control u for each x,t. The HJB equation becomes

$$-\partial_t J(x,t) \quad = \quad -\frac{1}{2R}(\partial_x J(x,t))^2 + V(x,t) + f(x,t)\partial_x J(x,t) + \frac{1}{2}\nu\partial_x^2 J(x,t)$$

Note, that after performing the minimization with respect to u, the HJB equation has become non-linear in J. We can, however, remove the non-linearity and this will turn out to greatly help us to solve the HJB equation. Define $\psi(x,t)$ through $J(x,t) = -\lambda \log \psi(x,t)$, with $\lambda = \nu R$ a constant. Then the HJB becomes

$$-\partial_t \psi(x,t) \quad = \quad \left(-\frac{V(x,t)}{\lambda} + f(x,t)\partial_x + \frac{1}{2}\nu\partial_x^2 \right) \psi(x,t) \tag{21}$$

Eq. 21 must be solved backwards in time with $\psi(x,t_f) = \exp(-\phi(x)/\lambda)$.

The linearity allows us to reverse the direction of computation, replacing it by a diffusion process, in the following way. Let $\rho(y,\tau|x,t)$ describe a diffusion process for $\tau > t$ defined by the Fokker-Planck equation

$$\partial_\tau \rho = -\frac{V}{\lambda}\rho - \partial_y(f\rho) + \frac{1}{2}\nu\partial_y^2\rho \tag{22}$$

with $\rho(y,t|x,t) = \delta(y-x)$.

Define $A(x,t) = \int dy \rho(y,\tau|x,t)\psi(y,\tau)$. It is easy to see by using the equations of motion Eq. 21 and 22 that $A(x,t)$ is independent of τ. Evaluating $A(x,t)$ for $\tau = t$ yields $A(x,t) = \psi(x,t)$. Evaluating $A(x,t)$ for $\tau = t_f$ yields $A(x,t) = \int dy \rho(y,t_f|x,t)\psi(x,t_f)$. Thus,

$$\psi(x,t) \quad = \quad \int dy \rho(y,t_f|x,t) \exp(-\phi(y)/\lambda) \tag{23}$$

We arrive at the important conclusion that the optimal cost-to-go $J(x,t) = -\lambda \log \psi(x,t)$ can be computed either by backward integration using Eq. 21 or by forward integration of a diffusion process given by Eq. 22. The optimal control is given by Eq. 20.

Both Eq. 21 and 22 are partial differential equations and, although being linear, still suffer from the curse of dimensionality. However, the great advantage of the forward diffusion process is that it can be simulated using standard sampling methods which can efficiently approximate these computations. In addition, as is discussed in [22], the forward diffusion process $\rho(y,t_f|x,t)$ can be written as a path integral and in fact Eq. 23 becomes a path integral. This path integral can then be approximated using standard methods, such as the Laplace approximation. Here however, we will focus on computing Eq. 23 by sampling.

As an example, we consider the control problem Eqs. 18 and 19 for the simplest case of controlled free diffusion:

$$V(x,t) = 0, \quad f(x,t) = 0, \quad \phi(x) = \frac{1}{2}\alpha x^2$$

In this case, the forward diffusion described by Eqs. 22 can be solved in closed form and is given by a Gaussian with variance $\sigma^2 = \nu(t_f - t)$:

$$\rho(y,t_f|x,t) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y-x)^2}{2\sigma^2}\right) \tag{24}$$

Since the end cost is quadratic, the optimal cost-to-go Eq. 23 can be computed exactly as well. The result is

$$J(x,t) \quad = \quad \nu R\log\left(\frac{\sigma}{\sigma_1}\right) + \frac{1}{2}\frac{\sigma_1^2}{\sigma^2}\alpha x^2 \tag{25}$$

with $1/\sigma_1^2 = 1/\sigma^2 + \alpha/\nu R$. The optimal control is computed from Eq. 20:

$$u = -R^{-1}\partial_x J = -R^{-1}\frac{\sigma_1^2}{\sigma^2}\alpha x = -\frac{\alpha x}{R + \alpha(t_f - t)}$$

We see that the control attracts x to the origin with a force that increases with t getting closer to t_f. Note, that the optimal control is independent of the noise ν. This is a general property of LQ control.

The path integral formulation

For more complex problems, we cannot compute Eq. 23 analytically and we must use either analytical approximations or sampling methods. For this reason, we write the diffusion kernel $\rho(y,t_f|x,t)$ in Eq. 23 as a path integral. The reason that we wish to do this is that this gives us a particular simple interpretation of how to estimate optimal control in terms of sampling trajectories.

For an infinitesimal time step ε, the probability to go from x to y according to the diffusion process Eq. 22 is given by the Gaussian distribution in y like Eq. 24 with $\sigma^2 = \nu\varepsilon$ and mean value $x + f(x,t)\varepsilon$. Together with the instantaneous decay rate

$\exp(-\varepsilon V(x,t)/\lambda)$, we obtain

$$\rho(y,t+\varepsilon|x,t) = \frac{1}{\sqrt{2\pi\nu\varepsilon}}\exp\left(-\frac{\varepsilon}{\lambda}\left[\frac{R}{2}\left(\frac{y-x}{\varepsilon}-f(x,t)\right)^2+V(x,t)\right]\right)$$

where we have used $\nu^{-1}=R/\lambda$.

We can write the transition probability as a product of n infinitesimal transition probabilities:

$$\begin{aligned}\rho(y,t_f|x,t) &= \int\int dx_1\ldots dx_{n-1}\rho(y,t_f|x_{n-1},t_{n-1})\ldots\rho(y_2,t_2|x_1,t_1)\rho(y_1,t_1|x,t)\\ &= \left(\frac{1}{\sqrt{2\pi\nu\varepsilon}}\right)^n\int dx_1\ldots dx_{n-1}\exp(-S_{\text{path}}(x_{0:n})/\lambda)\\ S_{\text{path}}(x_{0:n}) &= \varepsilon\sum_{i=0}^{n-1}\left[\frac{R}{2}\left(\frac{x_{i+1}-x_i}{\varepsilon}-f(x_i,t_i)\right)^2+V(x_i,t_i)\right]\end{aligned} \tag{26}$$

with $t_i=t+(i-1)\varepsilon$, $x_0=x$ and $x_n=y$.

Substituting Eq. 26 in Eq. 23 we can absorb the integration over y in the path integral and find

$$J(x,t) = -\lambda\log\left(\frac{1}{\sqrt{2\pi\nu\varepsilon}}\right)^n\int dx_1\ldots dx_n\exp\left(-\frac{1}{\lambda}S(x_{0:n})\right) \tag{27}$$

where

$$S(x_{0:n}) = \phi(x_n)+S_{\text{path}}(x_{0:n}) \tag{28}$$

is the Action associated with a path.

In the limit of $\varepsilon\to 0$, the sum in the exponent becomes an integral: $\varepsilon\sum_{i=0}^{n-1}\to\int_t^{t_f}d\tau$ and thus we can formally write

$$J(x,t) = -\lambda\log\int[dx]_x\exp\left(-\frac{1}{\lambda}S(x(t\to t_f))\right)+C \tag{29}$$

where the path integral $\int[dx]_x$ is over all trajectories starting at x and with $C\propto n\log n$ a diverging constant, which we can ignore because it does not depend on x and thus does not affect the optimal control. [3]

The path integral Eq. 27 is a log partition sum and therefore can be interpreted as a free energy. The partition sum is not over configurations, but over trajectories. $S(x(t\to t_f))$ plays the role of the energy of a trajectory and λ is the temperature. This link between stochastic optimal control and a free energy has two immediate consequences.

[3] The paths are continuous but non-differential and there are different forward are backward derivatives [29, 30]. Therefore, the continuous time description of the path integral and in particular $\dot{x}$ are best viewed as a shorthand for its finite n description.

1) Phenomena that allow for a free energy description, typically display phase transitions and spontaneous symmetry breaking. What is the meaning of these phenomena for optimal control? 2) Since the path integral appears in other branches of physics, such as statistical mechanics and quantum mechanics, we can borrow approximation methods from those fields to compute the optimal control approximately. First we discuss the small noise limit, where we can use the Laplace approximation to recover the PMP formalism for deterministic control [22]. Also, the path integral shows us how we can obtain a number of approximate methods: 1) one can combine multiple deterministic trajectories to compute the optimal stochastic control 2) one can use a variational method, replacing the intractable sum by a tractable sum over a variational distribution and 3) one can design improvements to the naive MC sampling.

The Laplace approximation

The simplest algorithm to approximate Eq. 27 is the Laplace approximation, which replaces the path integral by a Gaussian integral centered on the path that that minimizes the action. For each x_0 denote $x^*_{1:n} = \text{argmin}_{x_{1:n}} S(x_{0:n})$ the trajectory that minimizes the Action Eq. 28 and $x^* = (x_0, x^*_{1:n})$. We expand $S(x)$ to second order around x^* : $S(x) = S(x^*) + \frac{1}{2}(x-x^*)^T H(x^*)(x-x^*)$, with $H(x^*)$ the $n \times n$ matrix of second derivatives of S, evaluated at x^*. When we substitute this approximation for $S(x)$ in Eq. 27, we are left with a n-dimensional Gaussian integral, which we can solve exactly. The resulting optimal value function is then given by

$$J_{\text{laplace}}(x_0) = S(x^*) + \frac{\lambda}{2} \log \left(\frac{\nu \varepsilon}{\lambda} \right)^n \det H(x^*) \tag{30}$$

The control is computed through the gradient of J with respect to x_0. The second term, although not difficult to compute, has typically only a very weak dependence on x_0 and can therefore be ignored. In general, there may be more than one trajectory that is a local minimum of S. In this case, we use the trajectory with the lowest Action.

MC sampling

The stochastic evaluation of Eq. 23 consists of stochastic sampling of the diffusion process $\rho(y, t_f | x, t)$ with drift $f(x,t)dt$ and diffusion $d\xi$, and with an extra term due to the potential V. Whereas the other two terms conserve probability density, the potential term takes out probability density at a rate $V(x,t)dt/\lambda$. Therefore, the stochastic simulation of Eq. 22 is a diffusion that runs in parallel with the annihilation process:

$$\begin{aligned} dx &= f(x,t)dt + d\xi \\ x &= x + dx, \quad \text{with probability } 1 - V(x,t)dt/\lambda \\ x_i &= \dagger, \quad \text{with probability } V(x,t)dt/\lambda \end{aligned} \tag{31}$$

We can estimate $\rho(y, t_f | x, t)$ by running N times the diffusion process Eq. 31 from t to t_f using some fine discretization of time and initializing each time at $x(t) = x$. Denote

these N trajectories by $x_i(t \to t_f), i = 1, \ldots, N$. Then, $\psi(x,t)$ is estimated by

$$\hat{\psi}(x,t) = \frac{1}{N} \sum_{i \in \text{alive}} \exp(-\phi(x_i(t_f))/\lambda) \tag{32}$$

where 'alive' denotes the subset of trajectories that do not get killed along the way by the † operation. Note that, although the sum is typically over less than N trajectories, the normalization $1/N$ includes all trajectories in order to take the annihilation process properly into account.

From the path integral Eq. 27 we infer that there is another way to sample, which is sometimes preferable. The action contains a contribution from the drift and diffusion $\frac{R}{2}(\dot{x} - f)^2$, one from the potential V and one from the end cost ϕ. To correctly compute the path contributions, one can construct trajectories according to the drift, diffusion and V terms and assigns to each trajectory a cost $\exp(-\phi/\lambda)$, as we did in Eq. 32. Alternatively, one can construct trajectories according to the drift and diffusion terms only and assign to each trajectory a cost according to both V and ϕ in the following way.

Define the stochastic process

$$x = x + f(x,t)dt + d\xi \tag{33}$$

Then, $\psi(x,t)$ is also estimated by

$$\begin{aligned} \hat{\psi}(x,t) &= \frac{1}{N} \sum_{i=1}^{N} \exp\left(-S_{\text{cost}}(x_i(t \to t_f))/\lambda\right) \\ S_{\text{cost}}(x(t \to t_f)) &= \phi(x(t_f)) + \int_t^{t_f} d\tau V(x(\tau), \tau) \end{aligned} \tag{34}$$

The computation of u requires the gradient of $\psi(x,t)$ which can be computed numerically by computing ψ at nearby points x and $x \pm \delta x$ for some suitable value of δx.

The receding horizon problem

Up to now, we have considered a control problem with a fixed end time. In this case, the control explicitly depends on time as $J(x,t)$ changes as a function of time. Below, we will consider reinforcement learning, which is optimal control in a stationary environment with a discounted future reward cost. We can obtain similar behavior within the path integral control approach by considering a finite receding horizon. We consider a dynamics that does not explicitly depend on time $f(x,t) = f(x)$ and a stationary environment: $V(x,t) = V(x)$ and no end cost: $\phi(x) = 0$. Thus,

$$dx = (f(x) + u)dt + d\xi \tag{35}$$

$$C(x, u(t \to t+T)) = \left\langle \int_t^{t+T} dt \frac{R}{2} u(t)^2 + V(x(t)) \right\rangle_x \tag{36}$$

The optimal cost-to-go is given by

$$\begin{aligned} J(x) &= -\lambda \log \int dy \rho(y, t+T|x,t) \\ &= -\lambda \log \int [dx]_x \exp\left(-\frac{1}{\lambda} S_{\text{path}}(x(t \to t+T))\right) \end{aligned} \tag{37}$$

with ρ the solution of the Fokker-Planck equation Eq. 22 or S_{path} the Action given by Eq. 26.

Note, that because both the dynamics f and the cost V are time-independent, C does not explicitly depend on t. For the same reason, $\rho(y, t+T|x,t)$ and $J(x)$ do not depend on t. Therefore, if we consider a receding horizon where the end time $t_f = t+T$ moves with the actual time t, J gives the time-independent optimal cost-to-go to this receding horizon. The resulting optimal control is a time-independent function $u(x)$. The receding horizon problem is quite similar to the discounted reward problem of reinforcement learning.

REINFORCEMENT LEARNING

We now consider reinforcement learning, for which we consider a general stochastic dynamics given by a first order Markov process, that assigns a probability to the transition of x to x' under action u: $p_0(x'|x,u)$. We assume that x and u are discrete, as is usually done.

Reinforcement learning considers an infinite time horizon and rewards are discounted. This means that rewards in the far future contribute less than the same rewards in the near future. In this case, the optimal control is time-independent and consists of a mapping from each state to an optimal action. The treatment of this section is based in part on [19, 31].

We introduce a reward that depends on our current state x , our current action u and the next state x': $R(x,u,x')$. The expected reward when we take action u in state x is given as

$$R(x,u) = \sum_{x'} p_0(x'|x,u) R(x,u,x')$$

Note, that the reward is time-independent as is standard assumed in reinforcement learning.

We define a *policy* $\pi(u|x)$ as the conditional probability to take action u given that we are in state x. Given the policy π and given that we start in state x_t at time t, the probability to be in state x_s at time $s > t$ is given by

$$\begin{aligned} p_\pi(x_s; s|x_t; t) &= \sum_{u_{s-1}, x_{s-1}, \ldots, u_{t+1}, x_{t+1}, u_t} p_0(x_s|x_{s-1}, u_{s-1}) \ldots \\ &\quad \ldots \pi(u_{t+1}|x_{t+1}) p_0(x_{t+1}|x_t, u_t) \pi(u_t|x_t). \end{aligned}$$

Note, that since the policy is independent of time, the Markov process is stationary, i.e. $p_\pi(x'; t+s|x; t)$ is independent of t for any positive integer s, and we can write

$p_\pi(x';t+s|x;t) = p_\pi(x'|x;s-t)$. For instance

$$p_\pi(y;t+1|x,t) = \sum_u p_0(y|x,u)\pi(u|x) = p_\pi(y;t+2|x,t+1)$$

The *expected future discounted reward* in state x is defined as:

$$J_\pi(x) = \sum_{s=0}^{\infty} \sum_{x',u'} \pi(u'|x') p_\pi(x'|x;s) R(x',u') \gamma^s \tag{38}$$

with $0 < \gamma < 1$ the discount factor. J_π is also known as the value function for policy π. Note, that J_π only depends on the state and not on time. The objective of reinforcement learning is to find the policy π that maximizes J for all states. Simplest way to compute this is in the following way.

We can write a recursive relation for J_π in the same way as we did in the previous section.

$$\begin{aligned} J_\pi(x) &= \sum_u \pi(u|x) R(x,u) + \sum_{s=1}^{\infty} \sum_{x',u'} \pi(u'|x') p_\pi(x'|x;s) R(x',u') \gamma^s \\ &= \sum_u \pi(u|x) R(x,u) + \gamma \sum_{s=1}^{\infty} \sum_{x',u'} \sum_{x'',u''} \\ &\quad \pi(u'|x') p_\pi(x'|x'';s-1) p_0(x''|x,u'') \pi(u''|x) R(x',u') \gamma^{s-1} \\ &= \sum_{u,x'} \pi(u|x) p_0(x'|x,u) [R(x,u,x') + \gamma J_\pi(x')] = \sum_u \pi(u|x) A_\pi(x,u) \end{aligned} \tag{39}$$

where we have defined $A_\pi(x,u) = \sum_{x'} p_0(x'|x,u)[R(x,u,x') + \gamma J_\pi(x')]$. Given the time-independent policy π, complete knowledge of the environment p_0 and the reward function R, Eq. 39 gives a recursive equation for $J_\pi(x)$ in terms of itself. Solving for $J_\pi(x)$ by fixed point iteration is called *policy evaluation*: it evaluates the value of the policy π.

The idea of policy improvement is to construct a better policy from the value of the previous policy. Once we have computed J_π, we construct a new deterministic policy

$$\pi'(u|x) = \delta_{u,u(x)}, \qquad u(x) = \arg\max_u A_\pi(x,u) \tag{40}$$

π' is the deterministic policy to act *greedy* with respect to $A_\pi(x,u)$. For the new policy π' one can again determine the value $J_{\pi'}$ through policy evaluation. It can be shown (see [10]) that the solution for $J_{\pi'}$ is as least as good as the solution J_π in the sense that

$$J_{\pi'}(x) \geq J_\pi(x), \forall x$$

Thus, one can consider the following algorithm that starts with a random policy, computes the value of the policy through Eq.39, constructs a new policy through Eq. 40, constructs the value of that policy, etc, until convergence:

$$\pi^0 \to J_{\pi^0} \to \pi^1 \to J_{\pi^1} \to \pi^2 \ldots$$

One can show, that this procedure converges to a stationary value function $J^*(x)$ that is a fixed point of the above procedure. As we will show below, this fixed point is not necessary the global optimum because the policy improvement procedure can suffer from local minima.

The differences with the dynamic programming approach discussed before are that the optimal policy and the value function are time-independent in the case of reinforcement learning whereas the control and optimal cost-to-go are time dependent in the finite horizon problem. The dynamic programming equations are initiated at a future time and computed backwards in time. The policy evaluation equation is a fixed point equation and can be initialized with with an arbitrary value of $J_\pi(x)$.

TD learning and actor-critic networks

The above procedures assume that the environment in which the automaton lives is known. In particular Eq. 39 requires that both the environment $p_0(x'|x,u)$ and the reward $R(x,u,x')$ are known. When the environment is not known one can either first learn a model and then a controller or use a so-called model free approach, which yields the well-known TD(λ) and Q-learning algorithms.

When p_0 and R are not known, one can replace Eq. 39 by a sampling variant

$$J_\pi(x) = J_\pi(x) + \alpha(r + \gamma J_\pi(x') - J_\pi(x)). \tag{41}$$

with x the current state of the agent, x' the new state after choosing action u from $\pi(u|x)$ and r the actual observed reward. To verify that this stochastic update equation gives a solution of Eq. 39, look at its fixed point:

$$J_\pi(x) = R(x,u,x') + \gamma J_\pi(x').$$

This is a stochastic equation, because u is drawn from $\pi(u|x)$ and x' is drawn from $p_0(x'|x,\pi(x))$. Taking its expectation value with respect to u and x', we recover Eq. 39. Eq. 41 is the TD(0) algorithm [19]. The TD(λ) extension of this idea is to not only update state x but a larger set of recently visited states (eligibility trace) controlled by λ.

As in policy improvement, one can select a better policy from the values of the previous policy that is defined greedy with respect to to J_π. In principle, one should require full convergence of the TD algorithm under the policy π before a new policy is defined. However, full convergence takes a very long time, and one has the intuitive idea that also from a halfway converged value function one may be able to construct a new policy that may not be optimal, but at least better than the current policy. Thus, one can consider an algorithm where the updating of the value of the states, Eq. 41, and the definition of the policy, Eq. 40, are interleaved. The approach is known as actor-critic networks, where Eq. 41 is the critic that attempts to compute J_π to evaluate the quality of the current policy π, and where Eq. 40 is the actor that defines new policies based on the values J_π. [4]

[4] In mammals, the action of dopamine on striatal circuits has been proposed to implement such an actor-critic architecture [12], and recordings from monkey caudate neurons during simple associative

Q learning

A mathematically more elegant way to compute the optimal policy in a model free way is given by the Q learning algorithm [11]. Denote $Q(x,u)$ the optimal expected value of state x when taking action u and then proceeding optimally. That is

$$Q(x,u) = R(x,u) + \gamma \sum_{x'} p_0(x'|x,u) \max_{u'} Q(x',u') \tag{42}$$

and $J^*(x) = \max_u Q(x,u)$.

Its stochastic, on-line, version is

$$Q(x,u) = Q(x,u) + \alpha(R(x,u,x') + \gamma \max_{u'} Q(x',u') - Q(x,u)) \tag{43}$$

As before, one can easily verify that by taking the expectation value of this equation with respect to $p_0(x'|x,u)$ one recovers Eq. 42.

Note, that for this approach to work not only all states should be visited a sufficient number of times (as in the TD approach) but all state-action pairs. On the other hand, Q-learning does not require the policy improvement step and the repeated computation of value functions. Also in the Q-learning approach it is tempting to limit actions to those that are expected to be most successful, as in the TD approach, but this may again result in a suboptimal solution.

Both TD learning and Q learning require very long times to converge, which makes their application to artificial intelligence problems as well as to biological modeling problematic. Q learning works better in practice than TD learning. In particular, the choice of the relative update rates of the actor and critic in the TD approach can greatly affect convergence. There have been an number of approaches to speed up RL learning, in particular using hierarchical models where intermediate subgoals are formulated and learned, and function approximations, where the value is presented as a parametrized function and a limited number of parameters must be learned.

NUMERICAL EXAMPLES

Here we give some numerical examples of stochastic control. We first consider the delayed choice problem that illustrates the issue of symmetry breaking and timing in decision making. Subsequently we consider the receding horizon problem, both from the perspective of RL and from the path integral control point of view.

---

conditioning tasks signal an error in the prediction of future reward. [32] proposes that the function of these neurons is particularly well described by a specific class of reinforcement learning algorithms, and shows how a model that uses a dopamine-like signal to implement such an algorithm can learn to predict future rewards and guide action selection. More recent theoretical proposals have expanded the role of the dopamine signal to include the shaping of more abstract models of valuation [33, 34, 35]. It portrays the dopamine system as a critic whose influence extends beyond the generation of simple associative predictions to the construction and modification of complex value transformations.

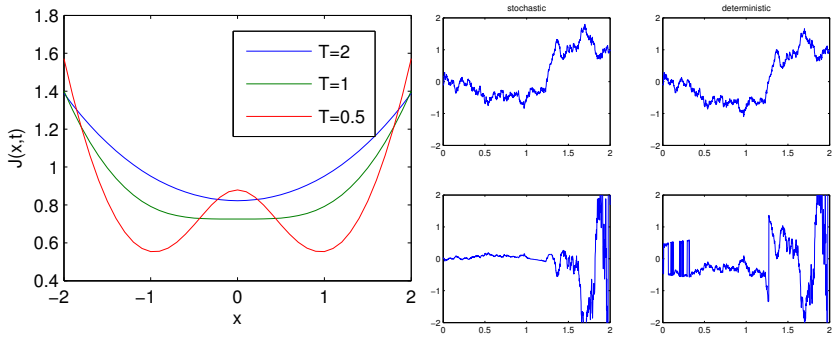

FIGURE 2. (Left) Symmetry breaking in J as a function of T implies a 'delayed choice' mechanism for optimal stochastic control. When the target is far in the future, the optimal policy is to steer between the targets. Only when $T < 1/\nu$ should one aim for one of the targets. $\nu = R = 1$. (Right) Sample trajectories (top row) and controls (bottom row) under stochastic control Eq. 44 (left column) and deterministic control Eq. 44 with $\nu = 0$ (right column), using identical initial conditions $x(t=0) = 0$ and noise realization.

The delayed choice

As a first example, we consider a dynamical system in one dimension that must reach one of two targets at locations $x = \pm 1$ at a future time t_f. As we mentioned earlier, the timing of the decision, that is *when* the automaton decides to go left or right, is the consequence of spontaneous symmetry breaking. To simplify the mathematics to its bare minimum, we take $V = 0$ and $f = 0$ in Eqs. 18 and 19 and $\phi(x) = \infty$ for all x, except for two narrow slits of infinitesimal size ε that represent the targets. At the targets we have $\phi(x = \pm 1) = 0$. In this simple case, we can compute J exactly (see [22]) and is given by

$$J(x,t) \quad = \quad \frac{R}{T}\left(\frac{1}{2}x^2 - \nu T \log 2\cosh\frac{x}{\nu T}\right) + \text{const.}$$

where the constant diverges as $\mathscr{O}(\log\varepsilon)$ independent of x and $T = t_f - t$ the time to reach the targets. The expression between brackets is a typical free energy with temperature νT. It displays a symmetry breaking at $\nu T = 1$ (fig. 2Left). For $\nu T > 1$ (far in the past or high noise) it is best to steer towards $x = 0$ (between the targets) and delay the choice which slit to aim for until later. The reason why this is optimal is that from that position the expected diffusion alone of size νT is likely to reach any of the slits without control (although it is not clear yet which slit). Only sufficiently late in time ($\nu T < 1$) should one make a choice. The optimal control is given by the gradient of J:

$$u = \frac{1}{T}\left(\tanh\frac{x}{\nu T} - x\right) \tag{44}$$

Figure 2Right depicts two trajectories and their controls under stochastic optimal control Eq. 44 and deterministic optimal control (Eq. 44 with $\nu = 0$), using the same realization of the noise. Note, that the deterministic control drives x away from zero to either one of the targets depending on the instantaneous value of $\text{sign}(x)$, whereas for large T the stochastic control drives x towards zero and is smaller in size. The stochastic control maintains x around zero and delays the choice for which slit to aim until $T \approx 1/\nu$.

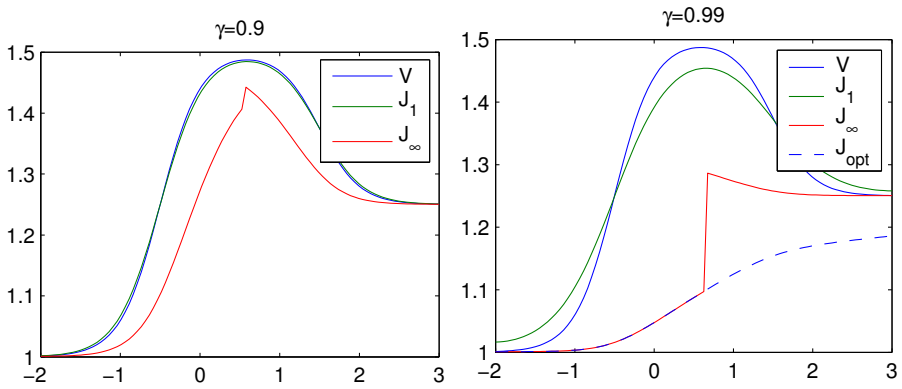

FIGURE 3. The policy improvement algorithm, that computes iteratively the value of a policy and then defines a new policy that is greedy with respect to this value function. In each figure, we show $V(x)$, the value $(1-\gamma)J_1(x)$ of the random initial policy, and $(1-\gamma)J_\infty(x)$ the value of the converged policy, all as a function of x.

The fact that symmetry breaking occurs in terms of the value of vT, is due to the fact that the action Eq. 26 $S_{\text{path}} \propto 1/T$, which in turn is due to the fact that we assumed $V = 0$. When $V \neq 0$, S_{path} will also contain a contribution that is proportional to T and the symmetry breaking pattern as a function of T can be very different.

Receding horizon problem

We now illustrate reinforcement learning and path integral control for a simple one dimensional example where the expected future reward within a discounted or receding horizon is optimized. The cost is given by V in figure 3 and the dynamics is simply moving to the left or the right.

For large horizon times, the optimal policy is to move from the local minimum to the global minimum of V (from right to left). The transient higher cost that is incurred by passing the barrier with high V is small compared to the long term gain of being in the global minimum instead of in the local minimum. For short horizon times the transient cost is too large and it is better to stay in the local minimum. We refer to these two qualitatively different policies as 'moving left' and 'staying put', respectively.

Reinforcement learning

In the case of reinforcement learning, the state space is discretized in 100 bins with $-2 < x < 3$. The action space is to move one bin to the left or one bin to the right: $u = \pm dx$. The dynamics is deterministic: $p_0(x'|x,u) = \delta_{x',x+u}$. The reward is given by $R(x,u,x') = -V(x')$, with $V(x)$ as given in figure 3. Reinforcement learning optimizes the expected discounted reward Eq. 38 with respect to π over all future contributions with discount factor γ. The discounting factor γ controls the effective horizon of the

rewards through $t_{\text{hor}} = -1/\log\gamma$. Thus for $\gamma \uparrow 1$, the effective horizon time goes to infinity.

We use the policy improvement algorithm, that computes iteratively the value of a policy and then defines a new policy that is greedy with respect to this value function. The initial policy is the random policy that assigns equal probability to move left or right.

For $\gamma = 0.9$, the results are shown in fig. 3Left. J_1 is the value of the initial policy. J_∞ is the value of the policy that is obtained after convergence of policy improvement. The asymptotic policy found by the policy improvement algorithm is unique, as is checked by starting from different initial policies, and thus corresponds to the optimal policy. From the shape of J_∞ one sees that the optimal policy for the short horizon time corresponding to $\gamma = 0.9$ is to 'stay put'.

For $\gamma = 0.99$, the results are shown in fig. 3Right. In this case the asymptotic policy found by policy improvement is no longer unique and depends on the initial policy. J_∞ is the asymptotic policy found when starting from the random initial policy and is suboptimal. J_{opt} is the value of the optimal policy (always move to the left) , which is clearly better since it has a lower value for all x. Thus, for $\gamma = 0.99$ the optimal policy is to 'move left'.

This phenomenon that policy improvement may find multiple suboptimal solutions persist for all larger values of γ (larger horizon times). We also ran Q-learning on the reinforcement learning task of fig. 3 and found the optimal policy for $\gamma = 0.9, 0.99$ and 0.999 (results not shown).

The number of value iterations of Eq. 39 depends strongly on the value of γ and empirically seem to scale proportional to $1/(1-\gamma)$ and thus can become quite large. The number of policy improvement steps in this simple example is only 1. The policy that is defined greedy with respect to J_1 is already within the discretization precision of the optimal policy. It has been checked that smoothing the policy updates ($\pi \leftarrow \alpha\pi + (1-\alpha)\pi_{\text{new}}$ for some $0 < \alpha < 1$) increases the number of policy improvement steps, but does not change fixed points of the algorithm.

Path integral control

We now compare reinforcement learning with the path integral control approach using a receding horizon time. The path integral control uses the dynamics Eq. 35 and cost Eq. 36 with $f(x) = 0$ and $V(x)$ as given in fig. 3. The solution is given by Eq. 37. This expression involves the computation of a high dimensional integral (one-dimensional paths) and is in general intractable. We use the MC sampling method and the Laplace approximation to find approximate solutions.

For the Laplace approximation of the cost-to-go, we use Eq. 30 and the result for short horizon time $T = 3$ is given by the dashed lines in fig. 4Middle and 4Right (identical curves). In fig. 4Left we show the minimizing Laplace trajectories for different initial values of x. This solution corresponds to the policy to 'stay put'. For comparison, we also show $TV(x)$, which is the optimal cost-to-go if V would be independent of x.

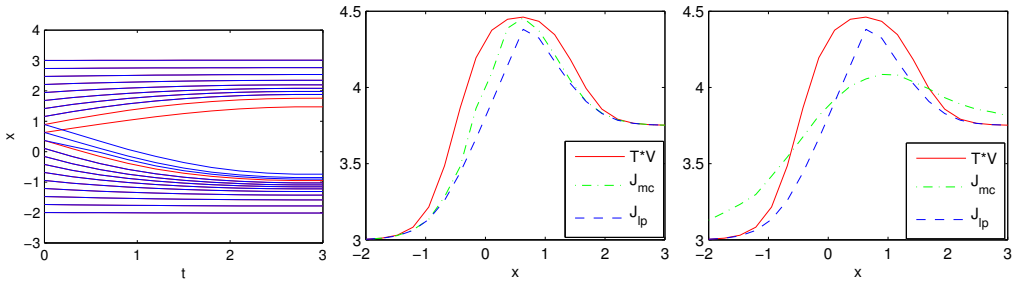

FIGURE 4. Left: Trajectories $x^*_{1:n}$ that minimize the Action Eq. 26 used in the Laplace approximation. $T = 3, R = 1$. Time discretization $dt = T/n, n = 10$. Middle: Optimal cost-to-go $J(x)$ for different x using the Laplace approximation (J_{lp}, dashed line) and the MC sampling (J_{mc}, dashed-dotted line) for $\nu = 0.01$. Right: idem for $\nu = 1$.

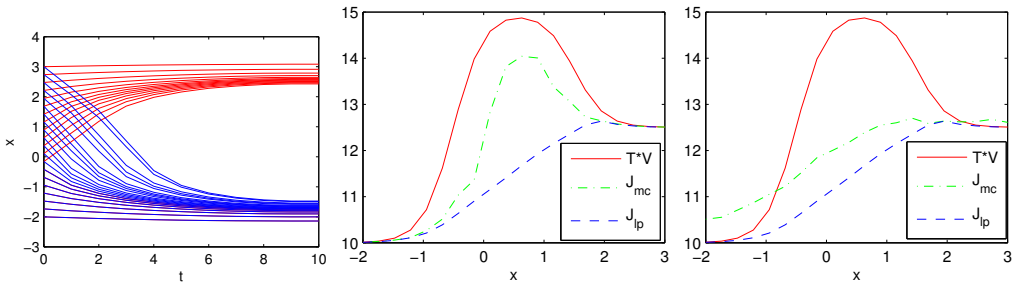

FIGURE 5. Left: Trajectories $x^*_{1:n}$ that minimize the Action Eq. 26 used in the Laplace approximation. $T = 10, R = 1$. Time discretization $dt = T/n, n = 10$. Middle: Optimal cost-to-go $J(x)$ for different x using the Laplace approximation (J_{lp}, dashed line) and the MC sampling (J_{mc}, dashed-dotted line) for $\nu = 0.01$. Right: idem for $\nu = 1$.

For a relatively large horizon time $T = 10$, the Laplace solution of the cost-to-to and the minimizing trajectories are shown in figure 5.

In figs. 4 and 5 we also show the results of the MC sampling (dashed dotted line). For each x, we sample $N = 1000$ trajectories according to Eq. 33 and estimate the cost-to-go using Eq. 34.

The Laplace approximation is accurate for low noise and becomes exact in the deterministic limit. It is a 'global' solution in the sense that the minimizing trajectory is minimal with respect to the complete (known) state space. Therefore, one can assume that the Laplace results for low noise in figs. 4Middle and 5Middle are accurate. In particular in the case of a large horizon time and low noise (fig. 5Middle), the Laplace approximation correctly proposes a policy to 'move left' whereas the MC sampler proposes (incorrectly) to 'stay put'.

The conditions for accuracy of the MC method are a bit more complex. The typical size of the area that is explored by the sampling process Eq. 33 is $x_{\text{mc}} = \sqrt{\nu T}$. In order for the MC method to succeed, this area should contain some of the trajectories that make the dominant contributions to the path integral. When $T = 3, \nu = 1, x_{\text{mc}} = 1.7$, which is sufficiently large to sample the dominant trajectories, which are the 'stay put' trajectories (those that stay in the local minima around $x = -2$ or $x = 3$). When $T = 10, \nu = 1$,

$x_{\rm mc} = 3.2$, which is sufficiently large to sample the dominant trajectories, which are the 'move left' trajectories (those that move from anywhere to the global minimum around $x = -2$). Therefore, for high noise we believe the MC estimates are accurate.

For low noise and a short horizon ($T = 3, \nu = 0.01$), $x_{\rm mc} = 0.17$ which is still ok to sample the dominant 'stay put'. However, for low noise and a long horizon ($T = 10, \nu = 0.01$), $x_{\rm mc} = 0.3$ which is too small to likely sample the dominant 'move left' trajectories. Thus, the MC sampler is accurate in three of these four cases (sufficiently high noise or sufficiently small horizon). For large horizon times and low noise the MC sampler fails.

Thus, the optimal control for short horizon time $T = 3$ is to 'stay put' more or less independent of the level of noise (fig. 4Middle $J_{\rm lp}$, fig. 4Right $J_{\rm mc}$). The optimal control for large horizon time $T = 10$ is to 'move left ' more or less independent of the level of noise (fig. 5Middle $J_{\rm lp}$, fig. 5Right $J_{\rm mc}$).

Note, that the case of a large horizon time corresponds to the case of γ close to 1 for reinforcement learning. We see that the results of RL and path integral control qualitatively agree.

Exploration

When the environment is not known, one needs to learn the environment. One can proceed in one of two ways: model-based or model-free. The model-based approach is simply to first learn the environment and then compute the optimal control. This optimal control computation is typically intractable but can be computed efficiently within the path integral framework. The model-free approach is to interleave exploration (learning the environment) and exploitation (behave optimally in this environment).

The model-free approach leads to the exploration-exploitation dilemma. The intermediate controls are optimal for the limited environment that has been explored, but are of course not the true optimal controls. These controls can be used to optimally exploit the known environment, but in general give no insight how to explore. In order to compute the truly optimal control for any point x one needs to know the whole environment. At least, one needs to know the location and cost of all the low lying minima of V. If one explores on the basis of an intermediate suboptimal control strategy there is no guarantee that asymptotically one will indeed explore the full environment and thus learn the optimal control strategy.

Therefore we conclude that control theory has in principle nothing to say about how to explore. It can only compute the optimal controls for future rewards once the environment is known. The issue of optimal exploration is not addressable within the context of optimal control theory. This statement holds for any type of control theory and thus also for reinforcement learning or path integral control.

There is one important exception to this, which is when one has some prior knowledge about the environment. There are two classes of prior knowledge that are considered in the literature. One is that the environment and the costs are smooth functions of the state variables. It is then possible to learn the environment using data from the known part of the environment only and extrapolate this model to the unknown parts

of the environment. One can then consider optimal exploration strategies relying on generalization.

The other type of prior knowledge is to assume that the environment and cost are drawn from some known probability distribution. An example is the k-armed bandit problem, for which the optimal exploration-exploitation strategy can be computed.

In the case of the receding horizon problem and path integral control, we propose naive sampling using the diffusion process Eq. 33 to explore states x and observe their costs $V(x)$. Note, that this exploration is not biased towards any control. We sample one very long trace at times $\tau = idt, i = 0, \ldots, N$, such that Ndt is long compared to the time horizon T. If at iteration i we are at a location x_i, we estimate $\psi(x_i, 0)$ by a single path contribution:

$$\psi(x_i, 0) = \exp\left(-\frac{dt}{\lambda} \sum_{j=i}^{j=i+n} V(x_j)\right) \tag{45}$$

with $T = ndt$ and $x_j, j = i+1, \ldots, i+n$ the n states visited after state x_i. We can compute this expression on-line by maintaining running estimates of $\psi(x_j)$ values of recently visited locations x_j. At iteration i we initialize $\psi(x_i) = 1$ and update all recently visited $\psi(x_j)$ values with the current cost:

$$\begin{aligned} \psi(x_i) &= 1 \\ \psi(x_j) &\leftarrow \psi(x_j) \exp\left(-\frac{dt}{\lambda} V(x_i)\right), \qquad j = i-n+2, \ldots, i-1 \end{aligned}$$

The results are shown in fig. 6 for the one-dimensional problem introduced in fig 3. We use a run of $N = 8000$ iterations, starting at $x = 0$. The diffusion process explores in expectation an area of size $\sqrt{\nu N dt} = 12.3$ around the starting value. From this one run, one can estimate simultaneously $J(x)$ for different horizon times ($T = 3$ and $T = 10$ in this case). Note, that these results are similar to the MC results in fig. 5.

By exploring the space according to Eq. 33, we can learn the environment. Once learned, we can use it to compute the optimal exploitation strategy as we discussed before. As we discussed before, we have no principled way to explore. Instead of using Eq. 33 we could choose any other random or deterministic method to decide at which points in space we want to compute the immediate cost and the expected cost-to-go. Our estimated model of the environment at time t can tell us how to best exploit it between t and $t+T$, but does not provide any information about how to explore those parts of the state space that have not yet been explored.

There is however, one advantage to use Eq. 33 for exploration, and that is that it not only explores the state space and teaches us about $V(x)$ at each of these states, but at the same time provides a large number of trajectories $x_{i:i+n}$ that we can use to compute the expected cost to go. If instead, we would sample x randomly one would require a second phase to estimate $\psi(x)$.

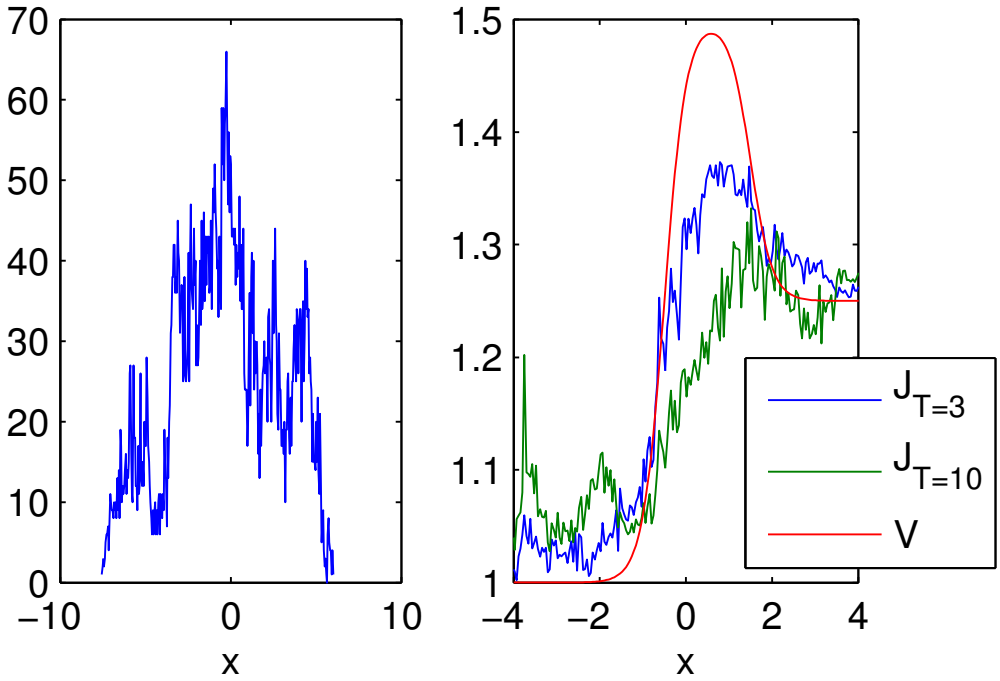

FIGURE 6. Sampling of $J(x)$ with one trajectory of $N = 8000$ iterations starting at $x = 0$. Left: The diffusion process Eq. 33 with $f = 0$ explores the area between $x = -7.5$ and $x = 6$. Shown is a histogram of the points visited (300 bins). In each bin x, an estimate of $\psi(x)$ is made by averaging all $\psi(x_i)$ with x_i from bin x (not shown). Right: $J_T(x)/T = -v \log \psi(x)/T$ versus x for $T = 3$ and $T = 10$ and $V(x)$ for comparison. Time discretization $dt = 0.02, v = 1, R = 1$.

A neural implementation

In this section, we propose a simple way to implement the control computation in a 'neural' way. It is well-known, that the brain represents the environment in terms of neural maps. These maps are topologically organized, in the sense that nearby neurons represent nearby locations in the environment. Examples of such maps are found in sensory areas as well as in motor areas. In the latter case, nearby neuron populations encode nearby motor acts.

Suppose that the environment is encoded in a neural map and let us consider a one-dimensional environment for simplicity. We also restrict to the receding horizon case with no end cost and no intrinsic dynamics: $f(x) = 0$. We consider a one-dimensional array of neurons, $i = 1, \ldots, m$ and denote the firing rate of the neurons at time t by $\rho_i(t)$. The brain structure encodes a simplified neural map in the sense that if the animal is at location $x = x_0 + idx$ in the external world, neuron i fires and all other neurons are quiet.

Normally, the activity in the neural map is largely determined by the sensory input, possibly augmented with a lateral recurrent computation. Instead, we now propose a dynamics that implements a type of thinking ahead or planning of the consequences of possible future actions. We assume that the neural array implements a space-discretized

version of the forward diffusion process as given by the Fokker-Planck Eq. 22:

$$\frac{d\rho_i}{dt} = -\frac{V_i}{\lambda}\rho_i(t) + \frac{\nu}{2}\sum_j D_{ij}\rho_j(t) \tag{46}$$

with D the diffusion matrix $D_{ii} = -2, D_{ii+1} = D_{ii-1} = 1$ and all other entries of D are zero. V_i is the cost, reward or risk of the environment at location i and must be know to the animal. Note, that each neuron can update its firing rate on the basis of the activity of itself and its nearest neighbors. Further, we assume that there is some additional inhibitory lateral connectivity in the network such that the total firing rate in the map is normalized: $\sum_i \rho_i(t) = 1$.

Suppose that at $t = 0$ the animal is at location x in the environment and wants to compute its optimal course of actions. Neuron i is active ($\rho_i(t = 0) = 1$) and all other neurons are quiet. By running the network dynamics from $t = 0$ to T in the absence of external stimuli, the animal can 'think' what will happen in the future.

For the environment of fig. 3 we illustrate this in fig. 7. The activity of the network is initialized as a sharply peaked Gaussian, centered on the actual location of the animal ($x = 1$). The figure shows $\rho(y, T|x = 1, 0)$ as a function of y for various horizon times T. For $T = 5$ the Gaussian has moved to the local minimum around $x = 2$. This means that for this horizon time the optimal course of action of the animal is to move to the right. For optimization of short-term reward, this is the nearest area of relative lower cost. When the network is run until $T = 10$, the peak around $x = 2$ disappears and a peak around $x = -1$ appears. For optimization of long-term reward, it is better to move to the global minimum, despite the fact that it is further away and requires a larger transient cost. Thus, by thinking ahead for a short or a long time, the animal can compute the actions that are optimal in a short or a long horizon, respectively. This is quite different from the reinforcement learning paradigm, where for each value of γ the Bellman equations should be solved.

In this very simple example, the decision whether to move left or right can be inferred simply from the mode of $\rho(y, T|x, 0)$. In general this does not need to be true and in any case, for the correct estimation of the size of the optimal control, the gradient of $\psi(x) = \int dy \rho(y, T|x, 0)$ must be computed.

Comparing RL and PI control

Let us here briefly summarize the main differences and similarities between reinforcement learning and path integral control. For a problem consisting of n states, RL requires the solution of a system of n recursive equations involving n rewards and n unknowns, which are the values at these n states (the Bellman equation). Through these equations, the value of each state depends thus on the value of each other state. Path integral control is different in the sense that the closed form solution Eq. 23 gives the value of each state in terms of all n rewards, but this can be computed independent from the value of all other states.

Computation time for PI control and RL both increase with the horizon time T. For RL, one empirically observes $t_{\text{cpu}} \propto 1/(1-\gamma)$ and if we define the horizon time as

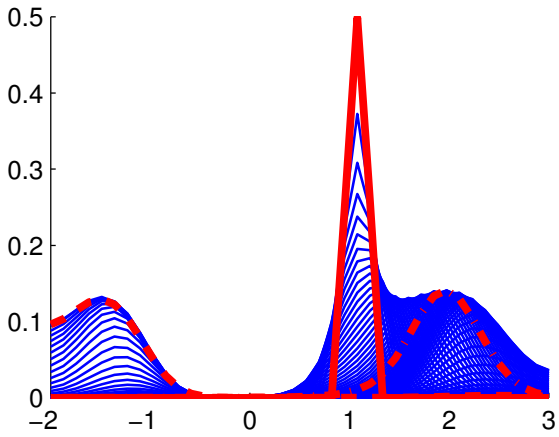

FIGURE 7. Thinking ahead. When the animal is at $x \approx 1$ it can start the dynamics eq. 46 to anticipate what will happen in the future. Top thin black line: $V(x)$ as before. Thin solid lines shows the time evolution of $\rho_i(t)$ from eq. 46. Thick solid, dot-dashed and dashed lines are $\rho_i(t)$ at $t = 0.1, 5$ and $t = 10$, respectively.

$T = -1/\log\gamma$ then $t_{\text{cpu}} \approx T$. For PI control, complexity is mainly determined by the time discretization dt of the diffusion computation and the trajectories. For instance, the Laplace approximation requires the minimization of an n dimensional function, with $T = ndt$, which due to the sparse structure of the Action can be done in $\mathcal{O}(n)$ time. The MC sampling requires a constant (possibly large) number of sampling trajectories each of length n and is therefore also proportional to n. The appropriate time discretization for large horizon times is not necessary the same as for small horizon times and therefore n may scale sub-linear with T.

In the case of RL, the computation of the value of the states depends on γ and for different γ the Bellman equations need to be solved separately. In the case of PI control, the solution for larger horizon time can be obtained by simply running the diffusion process for more time. The optimal control computation for the larger horizon time makes then effective use of the previously computed solution for shorter horizon time. For example, suppose that we know the solution for horizon times T: $\psi_T(x) = \int dy \rho_T(y|x)$. We can use this to compute a solution $\psi_{2T}(x) = \int dz \rho_{2T}(z|x) = \int dz dy \rho_T(z|y)\rho_T(y|x) = \int dy \psi_T(y)\rho_T(y|x)$.

With respect to exploration, RL and PI control are not very different. Both require to learn a model of the environment. In general, the control strategy that is optimal with respect to the partial environment that has been observed does not need to be a good strategy for exploration. If the objective is to learn a truly optimal control, the whole environment needs to be explored. When additional assumptions about the environment are made (for instance smoothness) this exploration can be made more efficient by relying on interpolation and extrapolation between observed states. Using the diffusion process Eq. 33 has the added advantage that it not only explores the full state space, but also estimates the optimal control from the explored sample trajectory. Extra criteria need to be considered (curiosity, surprise,...) to define the optimality of exploration.

DISCUSSION

In this paper, I have given an overview of the possible application of control theory to the modeling of animal behavior and learning. In the most general, and most interesting, case, stochastic optimal control is intractable and this has been a major obstacle for applications both in artificial intelligence and in biological modeling. Subsequently, I have introduced a class of non-linear stochastic control problems that can be efficiently solved using a path integral or by MC sampling. In this control formalism the central concept of cost-to-go becomes a free energy and methods and concepts from statistical physics can be readily applied. For instance the mean field and belief propagation methods can be used to approximate the free energy. An example of this is given in [23] in the context of multi-agent coordination.

I have discussed two types of control problems. Time-dependent problems where an intricate sequence of actions must be executed to reach a desired target. I have only described a very simple example where an agent must decide between to future targets and where due to the noise there is a non-trivial timing issue when to make this decision. The decision is made dynamically as the result of a spontaneous symmetry breaking of the cost-to-go.

The second problem is a time-independent problem where the expected future cost in a receding horizon has to be minimized. This problem is traditionally solved using reinforcement learning and I have compared that approach to the path integral approach. Both methods give more or less the same qualitative behavior as a function of the horizon time and there seems to be a rather mild dependence on the noise in the problem. I have indicated some of the computational advantages of the path integral approach

In all of this paper, we have assumed that the reward or cost is defined externally to the animal. At first sight, this seems quite acceptable. While the animal explores its environment, its initially more or less random sequences of actions will sometimes be rewarded positively (food, for instance) and sometimes negatively (starvation, danger). However, from the psychological literature [36] it is known that intrinsically-motivated behavior is essential for an organism to gain the competence necessary for autonomy. Intrinsic reward is related to achieving generic skill (options) that are useful components of externally rewarded tasks. For instance, a task that has external reward is to find food. Instead of learning this task with external reward only, it is commonly thought [36] that animals instead learn generic skills that then can later be used as components in tasks. Berlyne [37] suggests that the factors underlying intrinsic motivational effects involve novelty, surprise, incongruity, and complexity. He also hypothesized that moderate levels of novelty have the highest reward value and situations that are completely familiar (boredom) and completely unfamiliar (confusion) have lower reward. The combination of internal and external reward into a computational framework called options has been made by [38]. It is an open question how to incorporate such internal rewards in a more general control framework.

ACKNOWLEDGMENTS

This work is supported in part by the Dutch Technology Foundation and the BSIK/ICIS project.

REFERENCES

1. L. Abbott, J. Varela, K. Sen, and S. Nelson, *Science* pp. 220–224 (1997).
2. D. Blitz, K. Foster, and W. Regehr, *Nature Reviews Neuroscience* **5**, 630–640 (2004).
3. J. Gray, and P. Thompson, *Nature Reviews Neuroscience* **5** (2004).
4. W. T. Dickens, and J. R. Flynn, *Psychol. Rev.* **108**, 346–369 (2001).
5. D. Hebb, *The organization of behaviour*, Wiley, New York, 1949.
6. S. Kelso, A. Ganong, and T. Brouwn, *Proceedings National Academy of Science* **83**, 5326–5330 (1986).
7. P. Dayan, and L. Abbott, *Theoretical Neuroscience. Computational and Mathematical Modeling of Neural Systems*, MIT Press, New York, 2001.
8. R. Stengel, *Optimal control and estimation*, Dover publications, New York, 1993.
9. W. Fleming, and H. Soner, *Controlled Markov Processes and Viscosity solutions*, Springer Verlag, 1992.
10. R. Sutton, *Machine Learning* **3**, 9–44 (1988).
11. C. Watkins, *Learning from delayed rewards*, Ph.D. thesis, University of Cambridge, England (1989).
12. A. G. Barto, "," in *Models of Information Processing in the Basal Ganglia*, edited by J. C. Houk, J. L. Davis, and D. G. Beiser, MIT Press, Cambridge, Massachusetts, 1995, pp. 215–232.
13. R. Crites, and A. Barto, "Improving elevator performance using reinforcement learning," in *Advances in Neural Information Processing Systems 8: Proceedings of the 1995 Conference*, MIT Press, Cambridge MA, 1996.
14. S. Schaal, and C. Atkeson, *Control Systems Magazine* **14** (1994).
15. G. Tesauro, *Communications of the ACM* **38**, 58 – 68 (1995).
16. L. Sugrue, G. Corrado, and W. Newsome, *Nature Reviews Neuroscience* **6**, 365–375 (2005).
17. D. J. Barraclough, M. L. Conroy, and D. Lee, *Nature Neuroscience* **7**, 404–410 (2004).
18. W. Schultz, *Annu. Rev. Psychol.* **57**, 87–115 (2006).
19. R. Sutton, and A. Barto, *Reinforcement learning: an introduction*, MIT Press, 1998.
20. L. Pontryagin, V. Boltyanskii, R. Gamkrelidze, and E. Mishchenko, *The mathematical theory of optimal processes*, Interscience, 1962.
21. H. Kappen, *Physical Review Letters* **95**, 200201 (2005).
22. H. Kappen, *Journal of statistical mechanics: theory and Experiment* p. P11011 (2005).
23. B. Broek, W. W., and H. Kappen, *Journal of AI Research* (2006), in preparation.
24. W. Wiegerinck, B. v. d. Broek, and H. Kappen, "Stochastic optimal control in continuous space-time multi-agent systems," in *Proceedings UAI*, Association for Uncertainty in Artificial Intelligence, 2006, in press.
25. R. Bellman, and R. Kalaba, *Selected papers on mathematical trends in control theory*, Dover, 1964.
26. J. Yong, and X. Zhou, *Stochastic controls. Hamiltonian Systems and HJB Equations*, Springer, 1999.
27. W. Fleming, *Applied Math. Optim.* **4**, 329–346 (1978).
28. U. Jönsson, C. Trygger, and P. Ögren, Lectures on optimal control (2002).
29. E. Nelson, *Dynamical Theories of Brownian Motion*, Princeton University Press, Princeton, 1967.
30. F. Guerra, "Introduction to Nelson Stochastic mechanics as a Model for Quantum Mechanics," in *The Foundation of Quantum Mechanics*, Kluwer, Amsterdam, 1995.
31. L. Kaelbling, M. Littman, and A. Moore, *Journal of Artificial Intelligence research* **4**, 237–285 (1996).
32. W. Schultz, P. Dayan, and P. Montague, *Science* **275**, 1593–1598 (1997).
33. P. R. Montague, and G. S. Berns, *Neuron* **36**, 265–284 (2002).
34. S. M. McClure, N. D. Daw, and P. R. Montague, *Trends Neurosci.* **26**, 423–428 (2003).
35. P. R. Montague, S. E. Hyman, and J. D. Cohen, *Nature* **431**, 760–767 (2004).
36. R. White, *Psychological Review* **66**, 297–333 (1959).

37. D. E. Berlyne, *Conflict, Arousal. and Curiosity*, McGraw-Hill, New York, 1960.
38. S. Singh, A. Barto, and N. Chentanez, "Intrinsically motivated reinforcement learning," in *Advances in Neural Information Processing Systems 17: Proceedings of the 2004 Conference*, MIT Press, Cambridge MA, 2005.