

Efficient computation of optimal actions

Emanuel Todorov¹

Departments of Applied Mathematics and Computer Science & Engineering, University of Washington, Box 352420, Seattle, WA 98195

Edited by James L. McClelland, Stanford University, Stanford, CA, and approved April 28, 2009 (received for review November 16, 2007)

Optimal choice of actions is a fundamental problem relevant to fields as diverse as neuroscience, psychology, economics, computer science, and control engineering. Despite this broad relevance the abstract setting is similar: we have an agent choosing actions over time, an uncertain dynamical system whose state is affected by those actions, and a performance criterion that the agent seeks to optimize. Solving problems of this kind remains hard, in part, because of overly generic formulations. Here, we propose a more structured formulation that greatly simplifies the construction of optimal control laws in both discrete and continuous domains. An exhaustive search over actions is avoided and the problem becomes linear. This yields algorithms that outperform Dynamic Programming and Reinforcement Learning, and thereby solve traditional problems more efficiently. Our framework also enables computations that were not possible before: composing optimal control laws by mixing primitives, applying deterministic methods to stochastic systems, quantifying the benefits of error tolerance, and inferring goals from behavioral data via convex optimization. Development of a general class of easily solvable problems tends to accelerate progress—as linear systems theory has done, for example. Our framework may have similar impact in fields where optimal choice of actions is relevant.

action selection | cost function | linear Bellman equation | stochastic optimal control

If you are going to act, you might as well act in the best way possible. But which way is best? This is the general problem we consider here. Examples include a nervous system generating muscle activations to maximize movement performance (1), a foraging animal deciding which way to turn to maximize food (2), an internet router directing packets to minimize delays (3), an onboard computer controlling a jet engine to minimize fuel consumption (4), and an investor choosing transactions to maximize wealth (5). Such problems are often formalized as Markov decision processes (MDPs), with stochastic dynamics $p(x'|x, u)$ specifying the transition probability from state x to state x' under action u , and immediate cost $\ell(x, u)$ for being in state x and choosing action u . The performance criterion that the agent seeks to optimize is some cumulative cost that can be formulated in multiple ways. Throughout the article we focus on one formulation (total cost with terminal/goal states) and summarize results for other formulations.

Optimal actions cannot be found by greedy optimization of the immediate cost, but instead must take into account all future costs. This is a daunting task because the number of possible futures grows exponentially with time. What makes the task doable is the optimal cost-to-go function $v(x)$ defined as the expected cumulative cost for starting at state x and acting optimally thereafter. It compresses all relevant information about the future and thus enables greedy computation of optimal actions. $v(x)$ equals the minimum (over actions u) of the immediate cost $\ell(x, u)$ plus the expected cost-to-go $E[v(x')]$ at the next state x' :

$$v(x) = \min_u \{ \ell(x, u) + E_{x' \sim p(\cdot|x, u)} [v(x')] \}. \quad [1]$$

The subscript indicates that the expectation is taken with respect to the transition probability distribution $p(\cdot|x, u)$ induced by action u . Eq. 1 is fundamental to optimal control theory and is called the Bellman equation. It gives rise to Dynamic Programming (3) and

Reinforcement Learning (2) methods that are very general but can be inefficient. Indeed, Eq. 1 characterizes $v(x)$ only implicitly, as the solution to an unsolved optimization problem, impeding both analytical and numerical approaches.

Here, we show how the Bellman equation can be greatly simplified. We find an analytical solution for the optimal u given v , and then transform Eq. 1 into a linear equation. Short of solving the entire problem analytically, reducing optimal control to a linear equation is the best one can hope for. This simplification comes at a modest price: although we impose certain structure on the problem formulation, most control problems of practical interest can still be handled. In discrete domains our work has no precursors. In continuous domains there exists related prior work (6–8) that we build on here. Additional results can be found in our recent conference articles (9–11), online preprints (12–14), and supplementary notes [supporting information (SI) Appendix].

Results

Reducing Optimal Control to a Linear Problem. We aim to construct a general class of MDPs where the exhaustive search over actions is replaced with an analytical solution. Discrete optimization problems rarely have analytical solutions, thus our agenda calls for continuous actions. This may seem counterintuitive if one thinks of actions as symbols (“go left,” “go right”). However, what gives meaning to such symbols are the underlying transition probabilities—which are continuous. The latter observation is key to the framework developed here. Instead of asking the agent to specify symbolic actions, which are then replaced with transition probabilities, we allow the agent to specify transition probabilities $u(x'|x)$ directly. Formally, we have $p(x'|x, u) = u(x'|x)$.

Thus, our agent has the power to reshape the dynamics in any way it wishes. However, it pays a price for too much reshaping, as follows. Let $p(x'|x)$ denote the passive dynamics characterizing the behavior of the system in the absence of controls. The latter will usually be defined as a random walk in discrete domains and as a diffusion process in continuous domains. Note that the notion of passive dynamics is common in continuous domains but is rarely used in discrete domains. We can now quantify how “large” an action is by measuring the difference between $u(\cdot|x)$ and $p(\cdot|x)$. Differences between probability distributions are usually measured via Kullback–Leibler (KL) divergence, suggesting an immediate cost of the form

$$\ell(x, u) = q(x) + \text{KL}(u(\cdot|x) || p(\cdot|x)) = q(x) + E_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)} \right]. \quad [2]$$

The state cost $q(x)$ can be an arbitrary function encoding how (un)desirable different states are. The passive dynamics $p(x'|x)$ and controlled dynamics $u(x'|x)$ can also be arbitrary, except that

Author contributions: E.T. designed research, performed research, analyzed data, and wrote the paper.

The author declares no conflict of interest.

This article is a PNAS Direct Submission.

Freely available online through the PNAS open access option.

See Commentary on Page 11429.

¹E-mail: todorov@cs.washington.edu.

This article contains supporting information online at www.pnas.org/cgi/content/full/0710743106/DCSupplemental.

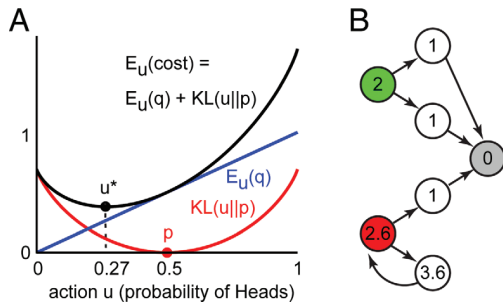


Fig. 1. New problem formulation. (A) A coin-toss example where the action corresponds to biasing the coin. The optimal bias is obtained by minimizing the sum (black) of the KL divergence cost (red) and the expected state cost (blue). Note that the tosses are independent and thus the temporal dynamics are irrelevant in this example. (B) A stochastic shortest-path problem illustrating how our framework can capture the benefits of error tolerance. See main text.

we require $u(x'|x) = 0$ whenever $p(x'|x) = 0$. This constraint is needed to make KL divergence well-defined. It has the added benefit of preventing the agent from jumping directly to goal states, and more generally from making state transitions that are physically impossible.

Fig. 1 illustrates the construction above with two simple examples. Fig. 1A is a coin-toss problem where $q(\text{Tails}) = 0, q(\text{Heads}) = 1$ and the passive dynamics correspond to an unbiased coin. The action u has the effect of biasing the coin. The optimal bias, which turns out to be $u^*(\text{Heads}) = 0.27$, achieves a trade-off between keeping the action cost and the expected state cost small. Note that the controller could have made the coin deterministic by setting $u(\text{Heads}) = 0$, but this is suboptimal because the associated action cost is too large. In general, the optimal actions resulting from our framework are stochastic. Fig. 1B is a shortest-path problem where $q = 0$ for the goal state (gray) and $q = 1$ for all other states. The passive dynamics correspond to the random walk on the graph. At the green state it does not matter which path is taken, so the optimal action equals the passive dynamics, the action cost is 0, and the cost-to-go (shown inside the circle) equals the length of the deterministic shortest path. At the red state, however, the optimal action deviates from the passive dynamics to cause a transition up, incurring an action cost of 0.6 and making the red state worse than the green state. In general, $v(x)$ is smaller when the task can be accomplished in multiple ways starting from x . This reflects a preference for error tolerance that is inherent in our framework.

We now return to the theoretical development. The results take on a simpler form when expressed in terms of the desirability function

$$z(x) = \exp(-v(x)). \quad [3]$$

This terminology reflects the fact that z is large at states where the cost-to-go v is small. Substituting Eq. 1 in Eq. 2, the Bellman equation can be written in terms of z as

$$-\log(z(x)) = q(x) + \min_u \left\{ E_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)z(x')} \right] \right\}. \quad [4]$$

The expression being minimized resembles KL divergence between u and pz , except that pz is not normalized to sum to 1. Thus, to obtain proper KL divergence (SI Appendix), we have to multiply and divide by the normalization term

$$\mathcal{G}[z](x) = \sum_{x'} p(x'|x)z(x') = E_{x' \sim p(\cdot|x)}[z(x')]. \quad [5]$$

Recall that KL divergence achieves its global minimum of zero when the 2 distributions are equal. Therefore, the optimal action u^* is proportional to pz :

$$u^*(x'|x) = \frac{p(x'|x)z(x')}{\mathcal{G}[z](x)}. \quad [6]$$

This represents the first general class of MDPs where the optimal actions can be found analytically given the optimal costs-to-go. Previously, such results were available only in continuous domains.

The Bellman equation can now be simplified (SI Appendix) by substituting the optimal action, taking into account the normalization term and exponentiating. The result is

$$z(x) = \exp(-q(x))\mathcal{G}[z](x). \quad [7]$$

The expectation $\mathcal{G}[z]$ is a linear operator; thus, Eq. 7 is linear in z . It can be written more compactly in vector notation. Enumerate the states from 1 to n , represent $z(x)$ and $q(x)$ with the n -dimensional column vectors \mathbf{z} and \mathbf{q} , and $p(x'|x)$ with the n -by- n matrix P , where the row-index corresponds to x and the column-index to x' . Then Eq. 7 becomes $\mathbf{z} = M\mathbf{z}$, where $M = \text{diag}(\exp(-\mathbf{q}))P$, \exp is applied element-wise and diag transforms vectors into diagonal matrices. The latter equation looks like an eigenvector problem, and indeed it can be solved (9) by using the power iteration method $\mathbf{z} \leftarrow M\mathbf{z}$ (which we call Z iteration). However, the problem here is actually simpler because the eigenvalue is 1 and $v(x) = q(x)$ at terminal states. If we define the index sets \mathcal{T} and \mathcal{N} of terminal and nonterminal states and partition \mathbf{z} , \mathbf{q} , and P accordingly, Eq. 7 becomes

$$(\text{diag}(\exp(\mathbf{q}_{\mathcal{N}})) - P_{\mathcal{N}\mathcal{N}})\mathbf{z}_{\mathcal{N}} = P_{\mathcal{N}\mathcal{T}} \exp(-\mathbf{q}_{\mathcal{T}}) \quad [8]$$

The unknown $\mathbf{z}_{\mathcal{N}}$ is the vector of desirabilities at the nonterminal states. It can be computed via matrix factorization or by using an iterative linear solver.

Let us now compare our result (Eq. 8) with policy iteration (3). We have to solve an equation of the form $A\mathbf{z} = \mathbf{b}$ just once. In policy iteration one has to solve an equation of the form $A_{(\pi)}\mathbf{v} = \mathbf{b}_{(\pi)}$ to evaluate the current policy π ; then, the policy has to be improved and the process repeated. Therefore, solving an optimal control problem in our formulation is computationally equivalent to half a step of policy iteration.

Thus far, we have studied MDPs with discrete state spaces. There exists a family of continuous (in space and time) problems related to our MDPs. These problems have stochastic dynamics

$$d\mathbf{x} = \mathbf{a}(\mathbf{x})dt + B(\mathbf{x})(\mathbf{u}dt + \sigma d\boldsymbol{\omega}). \quad [9]$$

$\boldsymbol{\omega}(t)$ is Brownian motion and σ is the noise amplitude. The cost rate is of the form

$$\ell(\mathbf{x}, \mathbf{u}) = q(\mathbf{x}) + \frac{1}{2\sigma^2} \|\mathbf{u}\|^2. \quad [10]$$

The functions $q(\mathbf{x})$, $\mathbf{a}(\mathbf{x})$, and $B(\mathbf{x})$ can be arbitrary. This problem formulation is fairly general and standard (but see Discussion). Consider, for example, a one-dimensional point with mass m , position x_p , and velocity x_v . Then, $\mathbf{x} = [x_p, x_v]^T$, $\mathbf{a}(\mathbf{x}) = [x_v, 0]^T$, and $B = [0, m^{-1}]^T$. The noise and control signals correspond to external forces applied to the point mass.

Unlike the discrete case where the agent could specify the transition probability distribution directly, here, \mathbf{u} is a vector that can shift the distribution given by the passive dynamics but cannot reshape it. Specifically, if we discretize the time axis in Eq. 9 with step h , the passive dynamics are Gaussian with mean $\mathbf{x} + h\mathbf{a}(\mathbf{x})$ and covariance $h\sigma^2 B(\mathbf{x})B(\mathbf{x})^T$, whereas the controlled dynamics are Gaussian with mean $\mathbf{x} + h\mathbf{a}(\mathbf{x}) + hB(\mathbf{x})\mathbf{u}$ and the same covariance. Thus, the agent in the continuous setting has less freedom compared with the discrete setting. Yet the two settings share many similarities, as follows. First, the KL divergence between the above Gaussians can be shown to be $\frac{h}{2\sigma^2} \|\mathbf{u}\|^2$, which is just the quadratic energy cost accumulated over time interval h . Second, given the

Table 1. Summary of results for all performance criteria

	Discrete	Continuous
Finite	$\exp(q)z_t = \mathcal{G}[z_{t+1}]$	$qz = \mathcal{L}[z] + \frac{\partial}{\partial t}z$
Total	$\exp(q)z = \mathcal{G}[z]$	$qz = \mathcal{L}[z]$
Average	$\exp(q - c)\tilde{z} = \mathcal{G}[\tilde{z}]$	$(q - c)\tilde{z} = \mathcal{L}[\tilde{z}]$
Discounted	$\exp(q)z = \mathcal{G}[z^\alpha]$	$qz = \mathcal{L}[z] - z \log(z^\alpha)$

optimal cost-to-go $v(\mathbf{x})$, the optimal control law can be computed analytically (4):

$$\mathbf{u}^*(\mathbf{x}) = -\sigma^2 B(\mathbf{x})^T v_{\mathbf{x}}(\mathbf{x}). \quad [11]$$

Here, subscripts denote partial derivatives. Third, the Hamilton–Jacobi–Bellman equation characterizing $v(\mathbf{x})$ becomes linear (6, 7) (SI Appendix) when written in terms of the desirability function $z(\mathbf{x})$:

$$q(\mathbf{x})z(\mathbf{x}) = \mathcal{L}[z](\mathbf{x}). \quad [12]$$

The second-order linear differential operator \mathcal{L} is defined as

$$\mathcal{L}[z](\mathbf{x}) = \mathbf{a}(\mathbf{x})^T z_{\mathbf{x}}(\mathbf{x}) + \frac{\sigma^2}{2} \text{trace}(B(\mathbf{x})B(\mathbf{x})^T z_{\mathbf{xx}}(\mathbf{x})). \quad [13]$$

Additional similarities between the discrete and continuous settings will be described below. They reflect the fact that the continuous problem is a special case of the discrete problem. Indeed, we can show (SI Appendix) that, for certain MDPs in our class, Eq. 7 reduces to Eq. 12 in the limit $h \rightarrow 0$.

The linear equations 7 and 12 were derived by minimizing total cost in the presence of terminal/goal states. Similar results can be obtained (SI Appendix) for all other performance criteria used in practice, in both discrete and continuous settings. They are summarized in Table 1. The constant c is the (unknown) average cost computed as the principal eigenvalue of the corresponding equation. \tilde{z} is the exponent of the differential cost-to-go function (3) (SI Appendix). The constant α is the exponential discount factor. Note that all equations are linear except in the discounted case. The finite-horizon and total-cost results in continuous settings were previously known (6, 7); the remaining six equations were derived in our work.

Applications. The first application is an algorithm for finding shortest paths in graphs (recall Fig. 1B). Let $s(x)$ denote the length of the shortest path from state x to the nearest terminal state. The passive dynamics p correspond to the random walk on the graph. The state costs are $q = \rho > 0$ for non-terminal states and $q = 0$ for terminal states. Let $v_\rho(x)$ denote the optimal cost-to-go function for given ρ . If the action costs were 0, the shortest path lengths would be $s(x) = \frac{1}{\rho} v_\rho(x)$ for all ρ . Here, the action costs are not 0, but nevertheless they are bounded. Because we are free to choose ρ arbitrarily large, we can make the state costs dominate the cost-to-go function, and so state costs dominate action cost.

$$s(x) = \lim_{\rho \rightarrow \infty} \frac{v_\rho(x)}{\rho}. \quad [14]$$

The construction above involves a limit and does not yield the shortest paths directly. However, we can obtain arbitrarily accurate (modulo numerical errors) approximations by setting ρ large enough. The method is illustrated in Fig. 2 on the graph of internet routers. There is a range of values of ρ for which all shortest path lengths are exactly recovered. Although a thorough comparison with dedicated shortest-path algorithms remains to be done, we suspect that they will not be as efficient as linear solvers. Problems with weighted edges can be handled with the general embedding method presented next.

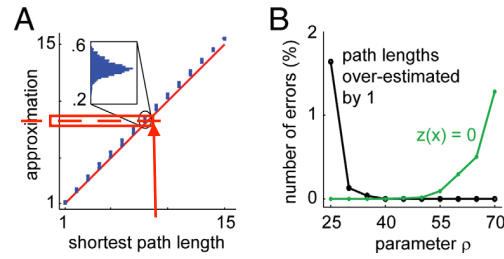


Fig. 2. Approximating shortest paths. The computation of shortest path lengths is illustrated here using the graph of internet routers and their connectivity as of 2003. This dataset is publicly available at www.caida.org. The graph has 190,914 nodes and 609,066 undirected edges. The shortest path length from each node to a specified destination node was computed exactly by using dynamic programming adapted to this problem, and also approximated by using our algorithm with $\rho = 40$. Our algorithm was ≈ 5 times faster, although both implementations were equally optimized. The exact shortest path lengths were integers between 1 and 15. (A) The approximate values were binned in 15 bins according to the corresponding correct value. The range of approximate values in each bin is shown with the blue symbols. The diagonal red line is the exact solution. (Inset) Histogram of all values in one bin, with the correct value subtracted. Note that all errors are between 0 and 1, thus rounding down to the nearest integer recovers the exact solution. This was the case for all bins. (B) To assess the effects of the free parameter ρ , we solved the above problem 500 times for each of 10 values of ρ between 25 and 70. In each instance of the problem, the set of destination nodes was generated randomly and had between 1 and 5 elements. The approximate shortest path lengths found by our algorithm were rounded down to the nearest integer and compared with the exact solution. The number of mismatches, expressed as a percent of the number of nodes and averaged over the 500 repetitions, is plotted in black. For large values of ρ the approximation becomes exact, as expected from Eq. 14. However, ρ cannot be set too large, because our algorithm multiplies by $\exp(-\rho)$, thus some elements of z may become numerically zero. The percentage of such numerical errors is plotted in green. There is a comfortable range of ρ where neither type of error is observed.

The second application is a method for continuous embedding of traditional MDPs with symbolic actions. It is reminiscent of linear programming relaxation in integer programming. Denote the symbolic actions in the traditional MDP with a , the transition probabilities with $\tilde{p}(x'|x, a)$, and the immediate costs with $\tilde{\ell}(x, a)$. We seek an MDP in our class such that for each (x, a) the action $\tilde{p}(\cdot|x, a)$ has cost $\tilde{\ell}(x, a)$: LP with constraints for all symbolic actions

$$q(x) + \text{KL}(\tilde{p}(\cdot|x, a) || p(\cdot|x)) = \tilde{\ell}(x, a). \quad [15]$$

For each x this yields a system of linear equations in q and $\log p$, which has a unique solution under a mild nondegeneracy condition (SI Appendix). If we keep the number of symbolic actions per state fixed while increasing the number of states (for example, by making a grid world larger and larger), the amount of computation needed to construct the embedding scales linearly with the number of states. Once the embedding is constructed, the optimal actions are computed and replaced with the nearest (in the sense of transition probabilities) symbolic actions. This yields an approximately optimal policy for the traditional MDP. We do not yet have theoretical error bounds but have found the approximation to be very accurate in practice. This method is illustrated in Fig. 3 on a machine repair problem adapted from ref. 3. The R^2 between the correct and approximate cost-to-go is 0.993.

The third application is a Monte Carlo method for learning z from experience in the absence of a model of the passive dynamics p and state costs q . Unlike the previous two applications, where we started with traditional MDPs and approximated them with MDPs in our class, here we assume that the problem is already in our class. The linear Bellman Eq. 7 can be unfolded recursively by replacing $z(x')$ with the expectation over the state following x'

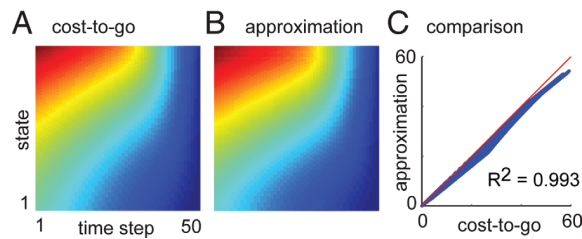


Fig. 3. Embedding traditional MDPs. The continuous embedding of traditional MDPs is illustrated here on a machine repair problem adapted from ref. 3. We have a machine whose state x_t at time t is an integer between 1 and 100. Larger x_t means that the machine is more broken and more costly to operate: we incur an operation cost of $0.02x_t$. There are 50 time steps (this is a finite-horizon problem). The state of the machine has a tendency to deteriorate over time: if we do nothing, x_{t+1} has a probability 0.9 of being one of $x_t \dots x_t + 8$ (chosen uniformly) and probability 0.1 of being one of $x_t - 9 \dots x_t - 1$. When x_t is near the edges we clip this distribution and renormalize it to sum to 1. The symbolic actions u_t are integers between 0 and 9 corresponding to how much we invest in repairing the machine in each time step. The repair cost is $0.1u_t$. The effect of the repairs is to circularly left-shift the above transition probability distribution by u_t positions. Thus, $u_t = 0$ corresponds to doing nothing; larger u_t causes larger expected improvement in the state of the machine. (A) The traditional MDP described above was embedded within our class, as outlined in the main text and described in detail in (SI Appendix). Then, $z_t(x)$ was computed and the approximation $-\log(z)$ to the optimal cost-to-go was plotted. Blue is small, red is large. (B) The traditional MDP was also solved by using dynamic programming and the optimal cost-to-go $v_t(x)$ was plotted in the same format as in A. (C) Scatter plot of the optimal versus approximate costs-to-go at 5,000 space-time points (blue). The R^2 between the two is 0.993, that is, the optimal values account for 99.3% of the variance in the approximate values. The red diagonal line corresponds to the ideal solution. We also computed (via extensive sampling) the performance of the optimal policy found by dynamic programming, the approximately optimal policy derived from our embedding, and a random policy. The performance of our approximation was 0.9% worse than optimal, whereas the performance of the random policy was 64% worse than optimal.

and so on, and then pushing all state costs inside the expectation operators. This yields the path-integral representation

$$z(x) = \mathbb{E}_{x_{t+1} \sim p(\cdot|x_t)} \left[\exp \left(- \sum_{t=0}^{t_f} q(x_t) \right) \right]. \quad [16]$$

with solved $z(x)$ and sampled $p(x)$, $u^*(x)$ can be calculated

$(x_0, x_1 \dots x_{t_f})$ are trajectories initialized at $x_0 = x$ and sampled from the passive dynamics, and t_f is the time when a goal state is first reached. A similar result holds in continuous settings, where the Feynman-Kac theorem states that the unique positive solution to Eq. 12 has a path-integral representation. The use of Monte Carlo methods for solving continuous optimal control problems was pioneered in ref. 7. Our result (Eq. 16) makes it possible to apply such methods to discrete problems with non-Gaussian noise.

The fourth application is related to the path-integral approach above but aims to achieve faster convergence. It is motivated by Reinforcement Learning (2) where faster convergence is often observed by using Temporal Difference (TD) methods. A TD-like method for our MDPs can be obtained from Eq. 7. It constructs an approximation \hat{z} using triplets (x_t, q_t, x_{t+1}) . Note that measuring the action costs is not necessary, \hat{z} is updated online as follows:

$$\hat{z}(x_t) \leftarrow (1 - \eta_t) \hat{z}(x_t) + \eta_t \exp(-q_t) \hat{z}(x_{t+1}). \quad [17]$$

η_t is a learning rate which decreases over time. We call this algorithm Z learning. Despite the resemblance to TD learning, there is an important difference. Our method learns the optimal cost-to-go directly, whereas TD methods are limited to learning the cost-to-go of a specific policy—which then needs to be improved, the cost-to-go relearned, and so on. Indeed Z learning is an off-policy method, meaning that it learns the cost-to-go for the optimal policy while sampling according to the passive dynamics. The only most important

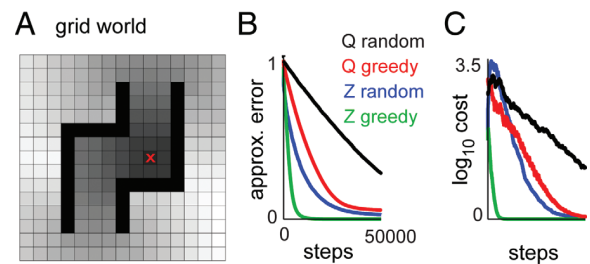


Fig. 4. Z learning and Q learning. Z learning and Q learning are compared in the context of a grid-world MDP. The goal state is marked "X." State transitions are allowed to all neighbors (including diagonal neighbors) and to the current state. Thus, there are at most 9 possible next states, less when the current state is adjacent to the obstacles shown in black or to the edges of the grid. p corresponds to a random walk. All state costs are $q = 1$ except for the goal state where $q = 0$. This MDP is within our class so Z learning can be applied directly. To apply Q learning we first need to construct a corresponding traditional MDP with symbolic actions. This is done as follows. For each state x we define a symbolic action with transition probability distribution matching the optimal $u^*(\cdot|x)$. We also define $N(x) - 1$ other symbolic actions, where $N(x)$ is the number of possible next states following x . Their transition probability distributions are obtained from $u^*(\cdot|x)$ by circular shifting; thus, they have the same shape as u^* but peak at different next states. All these symbolic actions incur cost $q(x) + \text{KL}(u^*(\cdot|x) || p(\cdot|x))$ matching the cost in our MDP. The resulting traditional MDP is guaranteed to have the same optimal cost-to-go as our MDP. (A) The grayscale image shows the optimal cost-to-go $v = -\log z$ where z is computed with our model-based method. Darker colors correspond to smaller values. Both Z learning and Q learning aim to approximate this function. (B) Error is defined as the average absolute difference between each approximation and the optimal costs-to-go, normalized by the average optimal cost-to-go. All approximations to z are initialized at 0. The learning rate decays as $\eta_t = c/(c+t)$, where $c = 5,000$ for Z learning and $c = 40,000$ for Q learning. Each simulation is repeated 10 times. The state is reset randomly whenever the goal state is reached. Each learning algorithm is tested by using a random policy corresponding to p , as well as a greedy policy. Q learning requires exploration; thus, we use an ϵ -greedy policy with $\epsilon = 0.1$. The values of c and ϵ are optimized manually for each algorithm. Z learning implicitly contains exploration so we can directly use the greedy policy, i.e., the policy $\hat{u}(x'|x)$ which appears optimal given the current approximation \hat{z} . Greedy Z learning requires importance sampling: the last term in Eq. 17 must be weighted by $p(x_{t+1}|x_t)/\hat{u}(x_{t+1}|x_t)$. Such weighting requires access to p . (C) Empirical performance of the policies resulting from each approximation method at each iteration. Z learning outperforms Q learning, and greedy methods outperform random sampling.

why greedy performs better than passive sampling????????

Reinforcement Learning method capable of doing this is Q learning (15). However, Q learning has the disadvantage of operating in the product space of states and actions, and is therefore less efficient. This is illustrated in Fig. 4 on a navigation problem.

The fifth application accelerates MDP approximations to continuous problems (16). Such MDPs are obtained via discretization and tend to be very large, calling for efficient numerical methods. Because continuous problems of the form (Eqs. 9 and 10) are limits of MDPs in our class, they can be approximated with MDPs in our class, which, in turn, are reduced to linear equations and solved efficiently. The same continuous problems can also be approximated with traditional MDPs and solved via dynamic programming. Both approximations converge to the same solution in the limit of infinitely fine discretization, and turn out to be equally accurate away from the limit, but our approximation is faster to compute. This is shown in Fig. 5 on a car-on-a-hill problem, where our method is ≈ 10 times faster than policy iteration and 100 times faster than value iteration.

The remaining applications have been developed recently. Below we summarize the key results and refer the reader to online preprints (12–14). The sixth application is a deterministic method for computing the most likely trajectory of the optimally controlled stochastic system. Combining Eqs. 6 and 7, the optimal control law can be written as $u^*(x'|x) = \exp(-q(x))p(x'|x) \frac{z(x')}{z(x)}$. Given a fixed

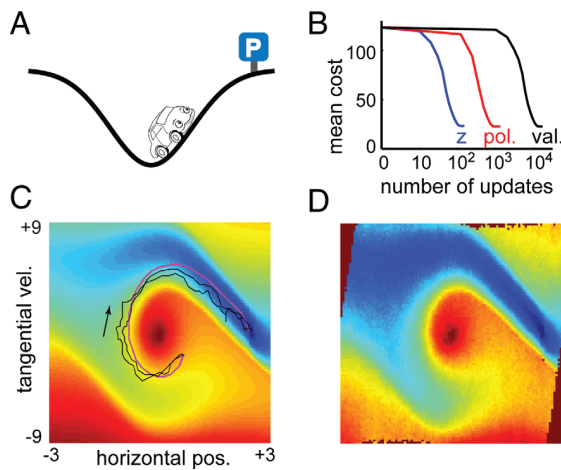


Fig. 5. Continuous problems. Comparison of our MDP approximation and a traditional MDP approximation on a continuous car-on-a-hill problem. (A) The car moves along a curved road in the presence of gravity. The control signal is tangential acceleration. The goal is to reach the parking lot with small velocity. (B) Z iteration (blue), policy iteration (red), and value iteration (black) converge to control laws with identical performance; however, Z iteration is 10 times faster than policy iteration and 100 times faster than value iteration. Note the log-scale on the horizontal axis. (C) The optimal cost-to-go for our approximation. Blue is small, red is large. The two black curves are stochastic trajectories resulting from the optimal control law. The thick magenta curve is the most likely trajectory of the optimally controlled stochastic system. It is computed by solving the deterministic optimal control problem described in the main text. (D) The optimal cost-to-go is inferred from observed state transitions by using our algorithm for inverse optimal control. Brown pixels correspond to states where we did not have data (i.e., no state transitions landed there); thus, the cost-to-go could not be inferred. The details are given in (SI Appendix).

initial state x_0 and final time T , the probability that the optimal control law u^* generates trajectory x_1, x_2, \dots, x_T is

$$\prod_{i=0}^{T-1} u^*(x_{i+1}|x_i) = \frac{z(x_T)}{z(x_0)} \prod_{i=0}^{T-1} \exp(-q(x_i)) p(x_{i+1}|x_i). \quad [18]$$

Omitting $z(x_0)$, which is fixed, and noting that $z(x_T) = \exp(-q(x_T))$, the negative log of Eq. 18 can be interpreted as the cumulative cost for a deterministic optimal control problem with immediate cost $q(x) - \log(p(x'|x))$. A related result is obtained in continuous settings when B is constant: the corresponding deterministic problem has dynamics $\dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}) + \mathbf{B}\mathbf{u}$ and cost $\ell(\mathbf{x}, \mathbf{u}) + \frac{1}{2} \text{div}(\mathbf{a}(\mathbf{x}))$. These results are important because optimal trajectories for deterministic problems can be found with efficient numerical methods (17) that avoid the curse of dimensionality. Our framework makes it possible to apply deterministic methods to stochastic control for the first time. The most likely trajectory in the car-on-a-hill problem is shown in Fig. 5C in magenta. See ref. 12 for details.

The seventh application exploits the duality between Bayesian estimation and stochastic optimal control. This duality is well-known for linear systems (4) and was recently generalized (8, 10) to nonlinear continuous systems of the form Eqs. 9 and 10. Duality also holds for our MDPs. Indeed, Eq. 18 can be interpreted as the posterior probability of a trajectory in a Bayesian smoothing problem, where $p(x_{i+1}|x_i)$ are the prior probabilities of the state transitions and $\exp(-q(x_i))$ are the likelihoods of some unspecified measurements. The desirability function in the control problem can be shown to be proportional to the backward filtering density in the dual estimation problem. Thus, stochastic optimal control problems can be solved by using Bayesian inference. See refs. 10 and 12 for details.

The eighth application concerns inverse optimal control, where the goal is to infer $q(x)$ given a dataset $\{x_k, x'_k\}_{k=1 \dots K}$ of state transitions generated by the optimal controller. Existing methods rely on guessing the cost function, solving the forward problem, comparing the solution with data, and improving the guess and iterating (18, 19). This indirect procedure can be inefficient when the forward problem is hard to solve. For our MDPs the inverse problem can be solved directly, by minimizing the convex function

$$L(\mathbf{v}) = \mathbf{d}^T \mathbf{v} + \mathbf{c}^T \log(P \exp(-\mathbf{v})), \quad [19]$$

where \mathbf{v} is the vector of (unknown) optimal costs-to-go, P is the passive dynamics matrix defined earlier, and \mathbf{d} and \mathbf{c} are the histograms of x'_k and x_k (i.e., d_i is the number of times that $x'_k = i$). It can be shown that $L(\mathbf{v})$ is the negative log-likelihood of the dataset. We have found empirically that its Hessian tends to be diagonally dominant, which motivates an efficient quasi-Newton method by using a diagonal approximation to the Hessian. Once the minimum is found, we can compute \mathbf{z} and then find \mathbf{q} from the linear Bellman equation. Fig. 5D illustrates this inverse optimal control method on the car-on-a-hill problem. See ref. 14 for details.

The ninth application is a method for constructing optimal control laws as combinations of certain primitives. This can be done in finite-horizon as well as total-cost problems with terminal/goal states, where the final cost plays the role of a boundary condition. Suppose we have a collection of component final costs $g_i(\mathbf{x})$ for which we have somehow computed the desirability functions $z_i(\mathbf{x})$. Linearity implies that, if the composite final cost $g(\mathbf{x})$ satisfies

$$\exp(-g(\mathbf{x})) = \sum_i w_i \exp(-g_i(\mathbf{x})) \quad [20]$$

for some set of w_i , then the composite desirability function is $z(\mathbf{x}) = \sum_i w_i z_i(\mathbf{x})$. This approach is particularly useful when the component problems can be solved analytically, as in the linear-quadratic-Gaussian (LQG) framework (4). In that case the component desirabilities $z_i(\mathbf{x})$ are Gaussians. By mixing them linearly, we can obtain analytical solutions to finite-horizon problems of the form Eqs. 9 and 10 where $\mathbf{a}(\mathbf{x})$ is linear, B is constant, $q(\mathbf{x})$ is quadratic; however, the final cost $g(\mathbf{x})$ is not constrained to be quadratic. Instead $g(\mathbf{x})$ can equal the negative log of any Gaussian mixture. This is a nontrivial extension to the widely used LQG framework. See ref. 13 for details.

Discussion

We formulated the problem of stochastic optimal control in a way which is rather general and yet affords substantial simplifications. Exhaustive search over actions was replaced with an analytical solution and the computation of the optimal cost-to-go function was reduced to a linear problem. This gave rise to efficient new algorithms speeding up the construction of optimal control laws. Furthermore, our framework enabled a number of computations that were not possible previously: solving problems with nonquadratic final costs by mixing LQG primitives, finding the most likely trajectories of optimally controlled stochastic systems via deterministic methods, solving inverse optimal control problems via convex optimization, quantifying the benefits of error tolerance, and applying off-policy learning in the state space as opposed to the state-action space.

These advances were made possible by imposing a certain structure on the problem formulation. First, the control cost must be a KL divergence—which reduces to the familiar quadratic energy cost in continuous settings. This is a sensible way to measure control energy and is not particularly restrictive. Second, the controls and the noise must be able to cause the same state transitions; the analog in continuous settings is that both the controls and the

noise act in the subspace spanned by the columns of $B(\mathbf{x})$. This is a more significant limitation. It prevents us from modeling systems subject to disturbances outside the actuation space. We are now pursuing an extension that aims to relax this limitation while preserving many of the appealing properties of our framework. Third, the noise amplitude and the control costs are coupled, and, in particular, the control costs are large when the noise amplitude is small. This can be compensated to some extent by increasing the state costs while ensuring that $\exp(-q(x))$ does not become numerically zero. Fourth, with regard to Z learning, following a policy other than the passive dynamics p requires importance-sampling correction based on a model of p . Such a model could presumably be learned online; however, this extension remains to be developed.

This framework has many potential applications and we hope that the list of examples will grow as other researchers begin to use it. Our current focus is on high-dimensional continuous problems such as those arising in biomechanics and robotics, where the discrete-time continuous-state MDP approximation is particularly promising. It leads to linear integral equations rather than differential equations, resulting in robust numerical methods. Furthermore it is well suited to handle discontinuities due to rigid-body collisions. Initial results using mixture-of-Gaussian approximations to the desirability function are encouraging (11), yet a lot more work remains.

ACKNOWLEDGMENTS. We thank Surya Ganguli, Javier Movellan, and Yuval Tassa for discussions and comments on the manuscript. This work was supported by the National Science Foundation.

1. Todorov E (2004) Optimality principles in sensorimotor control. *Nat Neurosci* 7(9):907–915.
2. Sutton R, Barto A (1998) *Reinforcement Learning: An Introduction* (MIT Press, Cambridge MA).
3. Bertsekas D (2001) *Dynamic Programming and Optimal Control* (Athena Scientific, Belmont, MA), 2nd Ed.
4. Stengel R (1994) *Optimal Control and Estimation* (Dover, New York).
5. Korn R (1997) *Optimal Portfolios: Stochastic Models for Optimal Investment and Risk Management in Continuous Time* (World Scientific, Teaneck, NJ).
6. Fleming W, Mitter S (1982) Optimal control and nonlinear filtering for nondegenerate diffusion processes. *Stochastics* 8:226–261.
7. Kappen HJ (2005) Linear theory for control of nonlinear stochastic systems. *Phys Rev Lett* 95:200201.
8. Mitter S, Newton N (2003) A variational approach to nonlinear estimation. *SIAM J Control Opt* 42:1813–1833.
9. Todorov E (2006) Linearly-solvable Markov decision problems. *Adv Neural Information Proc Syst* 19:1369–1376.
10. Todorov E (2008) General duality between optimal control and estimation. *IEEE Conf Decision Control* 47:4286–4292.
11. Todorov E (2009) Eigen-function approximation methods for linearly-solvable optimal control problems. *IEEE Int Symp Adapt Dynam Programm Reinforce Learning* 2:161–168.
12. Todorov E (2009) *Classic Maximum Principles and Estimation-Control Dualities for nonlinear Stochastic Systems*, preprint.
13. Todorov E (2009) *Compositionality of Optimal Control Laws*, preprint.
14. Todorov E (2009) *Efficient Algorithms for Inverse Optimal Control*, preprint.
15. Watkins C, Dayan P (1992) Q-learning. *Mach Learn* 8:279–292.
16. Kushner H, Dupuis P (2001) *Numerical Methods for Stochastic Optimal Control Problems in Continuous Time* (Springer, New York).
17. Bryson A, Ho Y (1969) *Applied Optimal Control* (Blaisdell Publishing Company, Waltham, MA).
18. Ng A, Russell S (2000) Algorithms for inverse reinforcement learning. *Int Conf Mach Learn* 17:663–670.
19. Liu K, Hertzmann A, Popovic Z (2005) Learning physics-based motion style with nonlinear inverse optimization. *ACM Trans Graphics* 24:1071–1081.

How can we learn efficiently to act optimally and flexibly?

Kenji Doya¹

Neural Computation Unit, Okinawa Institute of Science and Technology, Uruma, Okinawa 904-2234, Japan

When we walk to a shop in a town, we want to get there in the shortest time. However, finding the shortest route in a big city is quite tricky, because there are countless possible routes and the time taken for each segment of a route is uncertain. This is a typical problem of discrete optimal control, which aims to find the optimal sequence of actions to minimize the total cost from any given state to the goal state. The problems of optimal control are ubiquitous, from animal foraging to national economic policy, and there have been lots of theoretical studies on the topic. However, solving an optimal control problem requires a huge amount of computations except for limited cases. In this issue of PNAS, Emanuel Todorov (1) presents a refreshingly new approach in optimal control based on a novel insight as to the duality of optimal control and statistical inference.

The standard strategy in optimal control is to identify the “cost-to-go” function for each state, such as how much time you need from a street corner to your office. If such a cost-to-go function is available for all of the states, we can find the best route by simply following the nearest state with the lowest cost-to-go. More specifically, we use the formulation

$$\begin{aligned} &\text{minimal cost-to-go from one state} \\ &= \text{minimal (cost for one action} \\ &\quad + \text{cost-to-go from resulting state),} \end{aligned}$$

which is known as the “Bellman equation” (2). When there are n possible states, like n corners in your town, we have a system of n Bellman equations to solve. One headache in solving Bellman equations is the “minimal” operation. When there are many possible resulting states, because of randomness in state transition or choices of many possible actions, finding the minimal cost-to-go is not a trivial job. An easy solution has been known only for the case when the state transition is linear and the cost is a quadratic (second-order) function of the action and the state (3).

What is remarkable in Todorov’s proposal (1) is a wild reformulation of “action” and its cost. He recognizes the action as tweaking of the probability of the subsequent state and defines the action cost by the deviation of the tweaked state probability distribution from that with no action at all, called

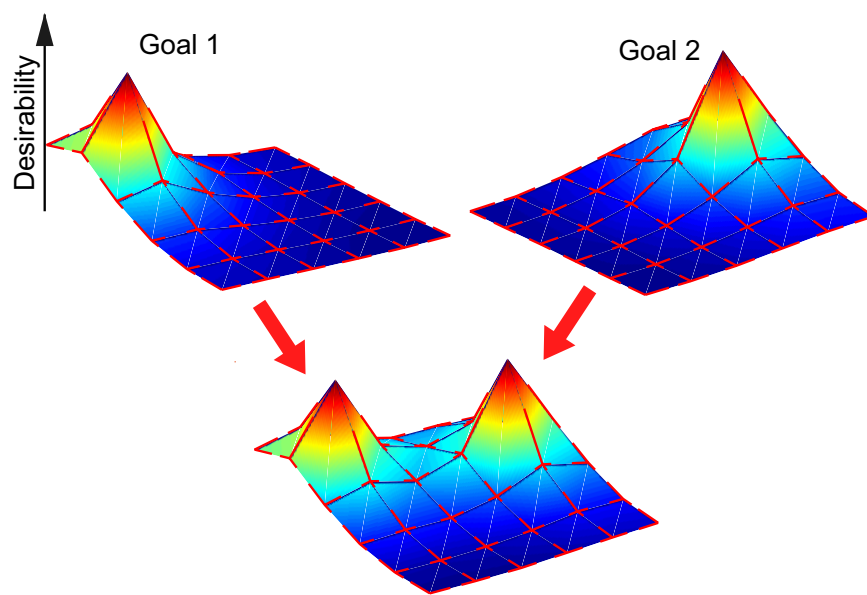


Fig. 1. Examples of desirability function in a task of city block navigation. An agent gains a reward (negative cost) of 1 by reaching to a particular corner (goal state), but pays a state cost of 0.1 for being in a nongoal corner and an action cost for deviating from random walk to one of adjacent corners. (Upper) Two examples of desirability functions with 2 different goal states. The desirability function has a peak at the goal state and serves as a guiding signal for navigation. The red segments on each corner show the optimal action, with the length proportional to the optimal probability of moving to that direction. (Lower) Shown is the desirability function when the reward is given at either goal position. In this case, the desirability function is simply the sum of the 2 desirability functions and the optimal action probability is the average of the 2 optimal actions probabilities weighted by the levels of 2 desirability functions at a given state. This compositionality allows flexible combination and selection of preacquired optimal actions depending on the given goal and the present state.

“passive dynamics.” Specifically, he takes so-called Kullback–Leibler divergence, which is the expected logarithmic ratio between the state distributions with an action and with passive dynamics. And in this particular setting, the minimization in the Bellman equation is achieved by reweighting the state distribution under the passive dynamics by the exponential of the sign-flipped cost-to-go function. This analytical form of minimization dramatically reduces the labor of solving the Bellman equation. Indeed, when we define the exponential of the sign-flipped cost-to-go function as the “desirability function,” the Bellman equation becomes

$$\begin{aligned} &\text{desirability of a state} \\ &= \text{exponential sign-flipped state cost} \\ &\quad \times \text{average desirability under} \\ &\quad \text{passive dynamics,} \end{aligned}$$

which is a linear equation. With the knowledge of the cost at each state and

the transition probability between the states under passive dynamics, the desirability function is given as an eigenvector of a matrix, which can be readily computed by common numerical software. Once the desirability function is derived, the optimal action is given by reweighting the state transition probability under passive dynamics in proportion to their desirability. Fig. 1 shows examples for desirability function and optimal actions in a simple shortest-time problem on a street grid.

One question is how widely this new formulation of action and action costs applies to real-world problems. In the article in this issue of PNAS (1) and related papers (4–6), Todorov has demonstrated that this principle can be extended to continuous-state, continuous-

Author contributions: K.D. wrote the paper.

The author declares no conflict of interest.

See companion article on page 11478.

¹E-mail: doya@oist.jp.

time optimal control problems and can be applied to a wide variety of tasks, including finding the shortest path in complex graphs, optimizing internet packet routing, and driving up a steep hill by an underpowered car.

Another issue is how to learn to act. In the standard formulation of optimal control, the cost or reward for being in a state, the cost for performing an action, and the probability of state transition depending on the action are explicitly given. However, in many realistic problems, such costs and transitions are not known a priori. Thus, we have to identify them before applying optimal control theory or take a short cut to learn to act based on actual experiences of costs and transitions. The latter way is known as reinforcement learning (7, 8). In Todorov's formulation (1), it is also possible to directly learn the desirability function without explicit knowledge of the costs and transitions, which he calls "Z-learning." The simulation result suggests that convergence Z-learning is considerably faster than the popular reinforcement learning algorithm called Q-learning (9). However, one problem with Z-learning is to find out the actual method of tweaking the state distribution as directed by the desirability function. It may be trivial in tasks like walking on grid-like streets, but may require another level of learning, for example, for shifting the body posture by stimulating hundreds of muscles.

Having a linear form of Bellman equation brings us another merit of compositionality of optimal actions (5). When there are two fairly good goals to achieve, what will be your optimal action? When the two goals are compatible, it may be a good idea to mix the actions for the two, but when they are far apart, you should make a crisp choice of which goal to aim at. For the linear form of Bellman equation, the boundary condition of the desirability function is specified by the cost at the goal states. Thus, once we calculate the desirability functions from boundary

conditions for a number of standard goals, we can derive the desirability function for a new goal if its cost or reward is expressed as a weighted combination of those for standard goals. The optimal action for the new composite goal takes an intuitive form: the optimal actions for component goals are weighted in proportion to the weights for the goals and the desirability at the present state as shown in Fig. 1 Lower. This desirability-weighted combination gives an elegant theoretical account of when actions can be mixed or should be crisply selected; it depends on the overlap of the desirability functions.

It is noteworthy that this new proposal (1) came from a researcher who has been working on the theory and experiments of human movement control (10, 11), where acting swiftly in the face of the delay and noise in sensory feedback poses a major challenge. This new formulation of optimal control is backed by a new insight of the duality between action and perception (6). In the world with noisy or delayed sensory inputs, finding the real present state of the world is not a trivial task. In the continuous domains, the Kalman filter (12) has been known as an optimal state estimator under linear dynamics, quadratic cost, and Gaussian noise, called the LQG setting. In the discrete domain, under the framework of hidden Markov models, many algorithms for state estimation have been developed in the field of machine learning research (13). It was almost a half-century ago when Kalman (12) pointed out the similarity between the equation for optimal state estimation by Kalman filter and the Bellman equation for optimal action in the LQG setting. Although this duality has been recognized as sheer coincidence or just theoretical beauty, studies in the brain mechanisms for perception and control led Todorov (6) to find the general duality between the computations for optimal action and optimal perception. The unusual definition of action cost by Kullback–Leibler diver-

gence in the new control scheme turns out to be quite natural when we recognize its duality with optimal state estimation in hidden Markov models.

With the favorable features of efficient solution and flexible combination, it is tempting to imagine if something similar could be happening in our brain. It has been proposed that human perception can be recognized as the process of Bayesian inference (14) and that they could be carried out in the neural circuit in the cerebral cortex (15, 16). By noting the duality between the computations for perception and action, it might be possible that, while the optimal sensory estimation is carried out in the sensory cortex, optimal control is implemented in the motor cortex or the frontal cortex. Neural activities for expected rewards, related to the cost-to-go function, have been found in the cerebral cortex and the subcortical areas including the striatum and the amygdala (8, 17–19). It will be interesting to test whether any neural representation of desirability function can be found anywhere in the brain. It is also interesting to think about whether off-line solution, like iterative computation of eigenvectors, and on-line solution, like Z-learning, can be implemented in the cortical or subcortical networks in the brain. There indeed is evidence that motor learning has both on-line and off-line components, the latter of which develops during resting or sleeping periods (20). It should also be possible to test whether human subjects or animals use desirability-weighted mixture and selection of actions in reaching for composite targets.

The series of works by Todorov (1, 4–6) is a prime example of a novel insight gained in the crossing frontlines of multidisciplinary research. It will have a wide impact on both theoretical and biological studies of action and perception.

ACKNOWLEDGMENTS. I am supported by a Grant-in-Aid for Scientific Research on Priority Areas "Integrative Brain Research" from the Ministry of Education, Culture, Sports, Science, and Technology of Japan.

- Todorov E (2009) Efficient computation of optimal actions. *Proc Natl Acad Sci USA* 106:11478–11483.
- Bertsekas D (2001) *Dynamic Programming and Optimal Control* (Athena Scientific, Belmont, MA), 2nd Ed.
- Bryson A, Ho Y (1969) *Applied Optimal Control* (Blaisdell, Waltham, MA).
- Todorov E (2009) Eigenfunction approximation methods for linearly-solvable optimal control problems. *IEEE International Symposium on Adaptive Dynamic Programming and Reinforcement Learning* (IEEE, Los Alamitos, CA) pp 1029.
- Todorov E (2009) Compositionality of optimal control laws. Preprint.
- Todorov E (2008) General duality between optimal control and estimation. *Proceedings of the 47th IEEE Conference on Decision and Control* (IEEE, Los Alamitos, CA), pp 4286–4292.
- Sutton R, Barto A (1998) *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA).
- Doya K (2007) Reinforcement learning: Computational theory and biological mechanisms. *HFSP J* 1:30–40.
- Watkins C, Dayan P (1992) Q-learning. *Machine Learning* 8:279–292.
- Todorov E, Jordan M (2002) Optimal feedback control as a theory of motor coordination. *Nat Neurosci* 5:1226–1235.
- Todorov E (2004) Optimality principles in sensorimotor control. *Nat Neurosci* 7:907–915.
- Kalman R (1960) A new approach to linear filtering and prediction problems. *ASME Trans J Basic Engineering* 82:35–45.
- Bishop C (2007) *Pattern Recognition and Machine Learning* (Springer, New York).
- Knill DC, Richards W (1996) *Perception as Bayesian Inference* (Cambridge Univ Press, Cambridge, UK).
- Rao RPN, Olshausen BA, Lewicki MS (2002) *Probabilistic Models of the Brain: Perception and Neural Function* (MIT Press, Cambridge, MA).
- Doya K, Ishii S, Pouget A, Rao RPN (2007) *Bayesian Brain: Probabilistic Approaches to Neural Coding* (MIT Press, Cambridge, MA).
- Platt ML, Glimcher PW (1999) Neural correlates of decision variables in parietal cortex. *Nature* 400:233–238.
- Samejima K, Ueda K, Doya K, Kimura M (2005) Representation of action-specific reward values in the striatum. *Science* 301:1337–1340.
- Paton JJ, Belova MA, Morrison SE, Salzman CD (2006) The primate amygdala represents the positive and negative value of visual stimuli during learning. *Nature* 439:865–870.
- Krakauer JW, Shadmehr R (2006) Consolidation of motor memory. *Trends Neurosci* 29:58–64.

Efficient computation of optimal actions

Supplementary notes

Emanuel Todorov

This supplement provides derivations of the results summarized in Table 1 in the main text, derivation of the relationship between the discrete and continuous formulations, and details on the MDP embedding method and the car-on-a-hill simulations.

1 Discrete problems

1.1 Generic formulation

We first recall the general form of a Markov decision process (MDP) and then introduce our new formulation which makes the problem more tractable. Consider a discrete set of states \mathcal{X} , a set $\mathcal{U}(x)$ of admissible actions at each state $x \in \mathcal{X}$, a transition probability function $p(x'|x, u)$, an instantaneous cost function $\ell(x, u)$, and (optionally) a final cost function $g(x)$ evaluated at the final state.

The objective of optimal control is to construct a control law $u = \pi_t^*(x)$ which minimizes the expected cumulative cost. Once a control law π is given the dynamics become autonomous, namely $x_{t+1} \sim p(\cdot|x_t, \pi_t(x_t))$. The expectations below are taken over state trajectories sampled from these dynamics. The expected cumulative cost for starting at state x and time t and acting according to π thereafter is denoted $v_t^\pi(x)$. This is called the cost-to-go function. It can be defined in multiple ways, as follows:

first exit total cost	$v^\pi(x) = \mathbb{E} \left[g(\underline{x_{t_f}}) + \sum_{\tau=0}^{t_f-1} \ell(x_\tau, \pi(x_\tau)) \right]$	
infinite horizon average cost	$v^\pi(x) = \lim_{t_f \rightarrow \infty} \frac{1}{t_f} \mathbb{E} \left[\sum_{\tau=0}^{t_f-1} \ell(x_\tau, \pi(x_\tau)) \right]$	
infinite horizon discounted cost	$v^\pi(x) = \mathbb{E} \left[\sum_{\tau=0}^{\infty} \alpha^\tau \ell(x_\tau, \pi(x_\tau)) \right]$	(1)
finite horizon total cost	$v_t^\pi(x) = \mathbb{E} \left[g(\underline{x_{t_f}}) + \sum_{\tau=\underline{t}}^{t_f-1} \ell(x_\tau, \pi_\tau(x_\tau)) \right]$	

In all cases the expectation is taken over trajectories starting at x . Only the finite horizon formulation allows explicit dependence on the time index. All other formulations are time-invariant, which is why the trajectories are initialized at time 0. The final time t_f is predefined in the finite horizon formulation. In the first exit formulation t_f is determined online as the time when a terminal/goal state $x \in \mathcal{A}$ is first reached. We can also think of the latter problem as being infinite horizon, assuming the system can remain forever in a terminal state without incurring extra costs.

The optimal cost-to-go function $v(x)$ is the minimal expected cumulative cost that any control law can achieve starting at state x :

$$v(x) = \min_{\pi} v^\pi(x) \tag{2}$$

The optimal control law is not always unique but the optimal cost-to-go is. The above minimum is achieved by the same control law(s) for all states x . This follows from Bellman's optimality principle, and has to do with the fact that the optimal action at state x does not depend on how we reached state x . The optimality principle also gives rise to the Bellman equation – which is a self-consistency condition satisfied by the optimal cost-to-go function. Depending on the definition of cumulative cost the Bellman equation takes on different forms, as follows:

first exit total cost	$v(x) = \min_{u \in \mathcal{U}(x)} \{ \ell(x, u) + \mathbb{E}_{x' \sim p(\cdot x, u)} [v(x')] \}, \quad v(x \in \mathcal{A}) = g(x)$	
infinite horizon average cost	$c + \tilde{v}(x) = \min_{u \in \mathcal{U}(x)} \{ \ell(x, u) + \mathbb{E}_{x' \sim p(\cdot x, u)} [\tilde{v}(x')] \}$	$\mathbb{E}[v(x)+c]=\mathbb{E}[v(x)]+c$
infinite horizon discounted cost	$v(x) = \min_{u \in \mathcal{U}(x)} \{ \ell(x, u) + \mathbb{E}_{x' \sim p(\cdot x, u)} [\alpha v(x')] \}$	α discount factor for infinite horizon
finite horizon total cost	$v_t(x) = \min_{u \in \mathcal{U}(x)} \{ \ell(x, u) + \mathbb{E}_{x' \sim p(\cdot x, u)} [v_{t+1}(x')] \}, \quad v_{t_f}(x) = g(x)$	

In the average cost formulation $\tilde{v}(x)$ has the meaning of a differential cost-to-go function, while c is the average cost which does not depend on the starting state. In the discounted cost formulation the constant $\alpha < 1$ is the exponential discount factor. In all formulations the Bellman equation involves minimization over the action set $\mathcal{U}(x)$. For generic MDPs such minimization requires exhaustive search. Our goal is to construct a class of MDPs for which this exhaustive search can be replaced with an analytical solution.

1.2 Restricted formulation where the Bellman equation is linear

In the traditional MDP formulation the controller chooses symbolic actions u which in turn specify transition probabilities $p(x'|x, u)$. In contrast, we allow the controller to choose transition probabilities $u(x'|x)$ directly, thus

$$p(x'|x, u) = u(x'|x) \quad (4)$$

The actions $u(\cdot|x)$ are real-valued vectors with non-negative elements which sum to 1. To prevent direct transitions to goal states, we define the passive or uncontrolled dynamics $p(x'|x)$ and require the actions to be compatible with it in the following sense:

$$\text{if } p(x'|x) = 0 \text{ then we require } u(x'|x) = 0 \quad (5)$$

Since $\mathcal{U}(x)$ is now a continuous set, we can hope to perform the minimization in the Bellman equation analytically. Of course this also requires a proper choice of cost function $\ell(x, u)$ and in particular proper dependence of ℓ on u :

$$\ell(x, u) = q(x) + \text{KL}(u(\cdot|x) || p(\cdot|x)) = q(x) + \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)} \right] \quad (6)$$

$q(x)$ can be an arbitrary function. At terminal states $q(x) = g(x)$. Thus, as far as the state cost is concerned, we have not introduced any restrictions. The control cost however must equal the Kullback-Leibler (KL) divergence between the controlled and passive dynamics. This is a natural way to measure how "large" the action is, that is, how much it pushes the system away from its default behavior.

With these definitions we can proceed to solve for the optimal actions given the optimal costs-to-go. In all forms of the Bellman equation the minimization that needs to be performed is

$$\min_{u \in \mathcal{U}(x)} \left\{ q(x) + \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)} \right] + \mathbb{E}_{x' \sim u(\cdot|x)} [w(x')] \right\} \quad (7)$$

where $w(x')$ is one of $v(x')$, $\check{v}(x')$, $\alpha v(x')$, $v_{t+1}(x')$. Below we give the derivation for $w = v$; the other cases are identical. The u -dependent expression being minimized in (7) is

$$\begin{aligned} \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)} \right] + \mathbb{E}_{x' \sim u(\cdot|x)} [v(x')] &= \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)} + v(x') \right] \\ &= \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x)} + \log \frac{1}{\exp(-v(x'))} \right] \\ &= \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x) \exp(-v(x'))} \right] \end{aligned} \quad (8)$$

The latter expression resembles KL divergence between u and $p \exp(-v)$, except that $p \exp(-v)$ is not normalized to sum to 1. In order to obtain a proper a KL divergence we introduce the normalization term

$$\mathcal{G}[z](x) = \sum_{x'} p(x'|x) z(x') = \mathbb{E}_{x' \sim p(\cdot|x)} [z(x')] \quad (9)$$

where the *desirability* function z is defined as

$$z(x) = \exp(-v(x)) \quad (10)$$

Now we multiply and divide the denominator on the last line of (8) by $\mathcal{G}[z](x)$. The derivation proceeds as follows:

$$\begin{aligned} \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x) z(x')} \right] &= \mathbb{E}_{x' \sim u(\cdot|x)} \left[\log \frac{u(x'|x)}{p(x'|x) z(x') \frac{\mathcal{G}[z](x)}{\mathcal{G}[z](x)}} \right] \\ &= \mathbb{E}_{x' \sim u(\cdot|x)} \left[-\log \mathcal{G}[z](x) + \log \frac{u(x'|x)}{p(x'|x) z(x') \frac{\mathcal{G}[z](x)}{\mathcal{G}[z](x)}} \right] \\ &= -\log \mathcal{G}[z](x) + \text{KL} \left(u(\cdot|x) \left\| \frac{p(\cdot|x) z(\cdot)}{\mathcal{G}[z](x)} \right\| \right) \end{aligned} \quad (11)$$

Thus the minimization involved in the Bellman equation takes the form

$$\min_{u \in \mathcal{U}(x)} \left\{ q(x) - \log \mathcal{G}[z](x) + \text{KL} \left(u(\cdot|x) \left\| \frac{p(\cdot|x) z(\cdot)}{\mathcal{G}[z](x)} \right\| \right) \right\} \quad (12)$$

$\min_u z(x)$ solves u with $z(x')$ as prior, which assumes that $z(x')$ in $g[z](x)$ is known

The first two terms do not depend on u . KL divergence achieves its global minimum of 0 if and only if the two probability distributions are equal. Thus the optimal action is

$$u^*(x'|x) = \frac{p(x'|x) z(x')}{\mathcal{G}[z](x)} \quad (13)$$

We can now drop the min operator, exponentiate the Bellman equations and write them in terms

of z as follows:

first exit total cost	$z(x) = \exp(-q(x)) \mathcal{G}[z](x)$	$\mathbf{z} = QP\mathbf{z}$
infinite horizon average cost	$\exp(-c) \tilde{z}(x) = \exp(-q(x)) \mathcal{G}[\tilde{z}](x)$	$\exp(-c) \tilde{\mathbf{z}} = QP\tilde{\mathbf{z}}$
infinite horizon discounted cost	$z(x) = \exp(-q(x)) \mathcal{G}[z^\alpha](x)$	$\mathbf{z} = QP\mathbf{z}^\alpha$
finite horizon total cost	$z_t(x) = \exp(-q(x)) \mathcal{G}[z_{t+1}](x)$	$\mathbf{z}_t = QP\mathbf{z}_{t+1}$

(14)

The third column gives the matrix form of these equations. The elements of the function $z(x)$ are assembled into the n -dimensional column vector \mathbf{z} , the passive dynamics $p(x'|x)$ are expressed as the n -by- n matrix P where the row index corresponds to x and the column index to x' , and Q is the n -by- n diagonal matrix with elements $\exp(-q(x))$ along its main diagonal. In the average cost formulation it can be shown that $\lambda = \exp(-c)$ is the principal eigenvalue. In the discounted cost formulation we have used the fact that $\exp(-\alpha v) = \exp(-v)^\alpha = z^\alpha$.

The optimal control law in the average cost, discounted cost and finite horizon cost formulations is again in the form (13), but z is replaced with \tilde{z} or z_{t+1} or z^α respectively.

2 Continuous problems

2.1 Generic formulation

As in the discrete case, we first summarize the generic problem formulation and then introduce a new formulation which makes it more tractable. Consider a controlled Ito diffusion of the form

$$d\mathbf{x} = \mathbf{f}(\mathbf{x}, \mathbf{u}) dt + F(\mathbf{x}, \mathbf{u}) d\omega \quad (15)$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is a state vector, $\mathbf{u} \in \mathbb{R}^{n_u}$ is a control vector, $\omega \in \mathbb{R}^{n_\omega}$ is standard multidimensional Brownian motion, \mathbf{f} is the deterministic drift term and F is the diffusion coefficient. If a control law $\mathbf{u} = \pi(\mathbf{x})$ is given the above dynamics become autonomous. We will need the 2nd-order linear differential operator $\mathcal{L}_{(\mathbf{u})}$ defined as

$$\mathcal{L}_{(\mathbf{u})}[v] = \mathbf{f}^\top v_{\mathbf{x}} + \frac{1}{2} \text{trace} \left(F F^\top v_{\mathbf{xx}} \right) \quad (16)$$

trace(FF^\top) = $\sum_i \sum_j F_{(ij)}^2 = \sum F_{(ij)}^2$ is the factor of second order in all directions of Taylor expansion

This is called the generator of the stochastic process (15). It is normally defined for autonomous dynamics, however the same notion applies for controlled dynamics as long as we make \mathcal{L} dependent on \mathbf{u} . The generator equals the expected directional derivative along the state trajectories. In the absence of noise (i.e. when $F = 0$) we have $\mathcal{L}_{(\mathbf{u})}[v] = \mathbf{f}^\top v_{\mathbf{x}}$ which is the familiar directional derivative. The trace term is common in stochastic calculus and reflects the noise contribution.

Let $g(\mathbf{x})$ be a final cost evaluated at the final time t_f which is either fixed or defined as a first exist time as before, and let $\ell(\mathbf{x}, \mathbf{u})$ be a cost rate. The expected cumulative cost resulting from

control law π can be defined in the following ways:

$$\begin{aligned}
\begin{array}{ll}
\text{first exit} & \\
\text{total cost} & v^\pi(\mathbf{x}) = \mathbb{E} \left[g(\mathbf{x}(t_f)) + \int_0^{t_f} \ell(\mathbf{x}(\tau), \pi(\mathbf{x}(\tau))) d\tau \right] \\
\text{infinite horizon} & \\
\text{average cost} & v^\pi(\mathbf{x}) = \lim_{t_f \rightarrow \infty} \frac{1}{t_f} \mathbb{E} \left[\int_0^{t_f} \ell(\mathbf{x}(\tau), \pi(\mathbf{x}(\tau))) d\tau \right] \\
\text{infinite horizon} & \\
\text{discounted cost} & v^\pi(\mathbf{x}) = \mathbb{E} \left[\int_0^\infty \exp(-\alpha\tau) \ell(\mathbf{x}(\tau), \pi(\mathbf{x}(\tau))) d\tau \right] \\
\text{finite horizon} & \\
\text{total cost} & v^\pi(\mathbf{x}, t) = \mathbb{E} \left[g(\mathbf{x}(t_f)) + \int_t^{t_f} \ell(\mathbf{x}(\tau), \pi(\mathbf{x}(\tau), \tau)) d\tau \right]
\end{array}
\end{aligned} \tag{17}$$

As in the discrete case, the optimal cost-to-go function is

$$v(\mathbf{x}) = \inf_{\pi} v^\pi(\mathbf{x}) \tag{18}$$

This function satisfies the Hamilton-Jacobi-Bellman (HJB) equation. The latter has different forms depending on the definition of cumulative cost, as follows:

$$\begin{aligned}
\begin{array}{ll}
\text{first exit} & \\
\text{total cost} & 0 = \min_{\mathbf{u}} \{ \ell(\mathbf{x}, \mathbf{u}) + \mathcal{L}_{(\mathbf{u})}[v](\mathbf{x}) \}, \quad v(\mathbf{x} \in \mathcal{A}) = g(\mathbf{x}) \\
\text{infinite horizon} & \\
\text{average cost} & c = \min_{\mathbf{u}} \{ \ell(\mathbf{x}, \mathbf{u}) + \mathcal{L}_{(\mathbf{u})}[\tilde{v}](\mathbf{x}) \} \\
\text{infinite horizon} & \\
\text{discounted cost} & \alpha v(\mathbf{x}) = \min_{\mathbf{u}} \{ \ell(\mathbf{x}, \mathbf{u}) + \mathcal{L}_{(\mathbf{u})}[v](\mathbf{x}) \} \\
\text{finite horizon} & \\
\text{total cost} & -v_t(\mathbf{x}, t) = \min_{\mathbf{u}} \{ \ell(\mathbf{x}, \mathbf{u}) + \mathcal{L}_{(\mathbf{u})}[v](\mathbf{x}, t) \}, \quad v(\mathbf{x}, t_f) = g(\mathbf{x})
\end{array}
\end{aligned} \tag{19}$$

Unlike the discrete case where the minimization over u had not been done analytically before, in the continuous case there is a well-known family of problems where analytical minimization is possible. These are problems with control-affine dynamics and control-quadratic costs:

$$\begin{aligned}
d\mathbf{x} &= (\mathbf{a}(\mathbf{x}) + B(\mathbf{x})\mathbf{u}) dt + C(\mathbf{x}) d\omega \\
\ell(\mathbf{x}, \mathbf{u}) &= q(\mathbf{x}) + \frac{1}{2} \mathbf{u}^\top R(\mathbf{x}) \mathbf{u}
\end{aligned} \tag{20}$$

For such problems the quantity $\ell(\mathbf{x}, \mathbf{u}) + \mathcal{L}_{(\mathbf{u})}[v](\mathbf{x})$ becomes quadratic in \mathbf{u} , and so the optimal control law can be found analytically given the gradient of the optimal cost-to-go:

$$\mathbf{u}^*(\mathbf{x}) = -R(\mathbf{x})^{-1} B(\mathbf{x})^\top v_{\mathbf{x}}(\mathbf{x}) \tag{21}$$

Substituting this optimal control law, the right hand side of all four HJB equations takes the form

$$q - \frac{1}{2} v_{\mathbf{x}}^\top B R^{-1} B^\top v_{\mathbf{x}} + \mathbf{a}^\top v_{\mathbf{x}} + \frac{1}{2} \text{tr} \left(C C^\top v_{\mathbf{xx}} \right) \tag{22}$$

where the dependence on \mathbf{x} (and t when relevant) has been suppressed for clarity. The latter expression is nonlinear in the unknown function v . Our goal is to make it linear.

2.2 Restricted formulation where the HJB equation is linear

As in the discrete case, linearity is achieved by defining the desirability function

$$z(\mathbf{x}) = \exp(-v(\mathbf{x})) \quad (23)$$

and rewriting the HJB equations in terms of z . To do so we need to express v and its derivatives in terms of z and its derivatives:

$$v = -\log(z), \quad v_{\mathbf{x}} = -\frac{z_{\mathbf{x}}}{z}, \quad v_{\mathbf{xx}} = -\frac{z_{\mathbf{xx}}}{z} + \frac{z_{\mathbf{x}}z_{\mathbf{x}}^{\top}}{z^2} \quad (24)$$

The last equation is key, because it contains the term $z_{\mathbf{x}}z_{\mathbf{x}}^{\top}$ which will cancel the nonlinearity present in (22). Substituting (24) in (22), using the properties of the trace operator and rearranging yields

$$q - \frac{1}{z} \left(\mathbf{a}^{\top} z_{\mathbf{x}} + \frac{1}{2} \text{tr} \left(CC^{\top} z_{\mathbf{xx}} \right) \right) + \frac{1}{2z} z_{\mathbf{x}}^{\top} B R^{-1} B^{\top} z_{\mathbf{x}} - \frac{1}{2z} z_{\mathbf{x}}^{\top} C C^{\top} z_{\mathbf{x}} \quad (25)$$

Now we see that the nonlinear terms cancel when $CC^{\top} = BR^{-1}B^{\top}$, which holds when

$$C(\mathbf{x}) = B(\mathbf{x}) \sqrt{R(\mathbf{x})}^{-1} \quad (26)$$

The s.p.d. matrix square root of R^{-1} is uniquely defined because R is s.p.d. Note that in the main text we assumed $R = I/\sigma^2$ and so $C = B\sigma$. The present derivation is more general. However the noise and controls still act in the same subspace, and the noise amplitude and control cost are still inversely related.

Assuming condition (26) is satisfied, the right hand side of all four HJB equations takes the form

$$q - \frac{1}{z} \mathcal{L}_{(0)}[z] \quad (27)$$

where $\mathcal{L}_{(0)}$ is the generator of the passive dynamics (corresponding to $\mathbf{u} = 0$). We will omit the subscript (0) for clarity. For this class of problems the generator of the passive dynamics is

$$\mathcal{L}[z] = \mathbf{a}^{\top} z_{\mathbf{x}} + \frac{1}{2} \text{tr} \left(CC^{\top} z_{\mathbf{xx}} \right) \quad (28)$$

Multiplying by $-z \neq 0$ we now obtain the transformed HJB equations

first exit total cost	$0 = \mathcal{L}[z] - qz$	
infinite horizon average cost	$-c\tilde{z} = \mathcal{L}[\tilde{z}] - q\tilde{z}$	
infinite horizon discounted cost	$z \log(z^{\alpha}) = \mathcal{L}[z] - qz$	(29)
finite horizon total cost	$-z_t = \mathcal{L}[z] - qz$	

As in the discrete case, these equations are linear in all but the discounted cost formulation.

3 Relation between the discrete and continuous formulations

Here we show how the continuous formulation can be obtained from the discrete formulation. This will be done by first making the state space of the MDP continuous (which merely replaces the sums with integrals), and then taking a continuous-time limit. Recall that the passive dynamics in the continuous formulation are

$$d\mathbf{x} = \mathbf{a}(\mathbf{x}) dt + C(\mathbf{x}) d\omega \quad (30)$$

Let $p_{(h)}(\cdot|\mathbf{x})$ denote the transition probability distribution of (30) over a time interval h . We can now define an MDP in our class with passive dynamics $p_{(h)}(\cdot|\mathbf{x})$ and state cost $hq(\mathbf{x})$. Let $z_{(h)}(\mathbf{x})$ denote the desirability function for this MDP, and suppose the following limit exists:

$$s(\mathbf{x}) = \lim_{h \rightarrow 0} z_{(h)}(\mathbf{x}) \quad (31)$$

The linear Bellman equation for the above MDP is

$$z_{(h)}(\mathbf{x}) = \exp(-hq(\mathbf{x})) E_{\mathbf{x}' \sim p_{(h)}(\cdot|\mathbf{x})} [z_{(h)}(\mathbf{x}')] \quad (32)$$

Our objective now is to take the continuous-time limit $h \rightarrow 0$ and recover the PDE

$$qs = \mathcal{L}[s] \quad (33)$$

A straightforward limit in (32) yields the trivial result $s = s$ because $p_{(0)}(\cdot|\mathbf{x})$ is the Dirac delta function centered at \mathbf{x} . However we can rearrange (32) as follows:

$$\frac{\exp(hq(\mathbf{x})) - 1}{h} z_{(h)}(\mathbf{x}) = \frac{E_{\mathbf{x}' \sim p_{(h)}(\cdot|\mathbf{x})} [z_{(h)}(\mathbf{x}') - z_{(h)}(\mathbf{x})]}{h} \quad (34)$$

$q = \lim_{h \rightarrow 0} (\exp(hq(\mathbf{x})) - 1)/h$
 $s = \lim_{h \rightarrow 0} z_{(h)}(\mathbf{x})$

The limit of the left hand side now yields qs . The limit of the right hand side closely resembles the generator of the passive dynamics (i.e. the expected directional derivative). If we had s instead of $z_{(h)}$ that limit would be exactly $\mathcal{L}[s]$ and we would recover (33). The same result is obtained by assuming that $z_{(h)}$ converges to s sufficiently rapidly and uniformly, so that

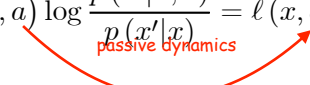
$$E_{\mathbf{x}' \sim p_{(h)}(\cdot|\mathbf{x})} [z_{(h)}(\mathbf{x}') - z_{(h)}(\mathbf{x})] = E_{\mathbf{x}' \sim p_{(h)}(\cdot|\mathbf{x})} [s(\mathbf{x}') - s(\mathbf{x})] + o(h^2) \quad (35)$$

Then the limit of (34) yields (33).

4 Embedding of traditional MDPs

In the main text we outlined a method for embedding traditional MDPs. The details are provided here. Denote the symbolic actions in the traditional MDP with a , the transition probabilities with $\tilde{p}(x'|x, a)$ and the costs with $\tilde{\ell}(x, a)$. We seek an MDP within our class such that for each (x, a) the action $u^a(\cdot|x) = \tilde{p}(\cdot|x, a)$ has cost $\ell(x, u^a) = \tilde{\ell}(x, a)$. In other words, for each symbolic action in the traditional MDP we want a corresponding continuous action with the same cost and transition probability distribution. The above requirement for proper embedding is equivalent to

$$q(x) + \sum_{x'} \tilde{p}(x'|x, a) \log \frac{\tilde{p}(x'|x, a)}{p(x'|x)} = \tilde{\ell}(x, a), \quad \forall x \in \mathcal{X}, a \in \tilde{\mathcal{U}}(x) \quad (36)$$



 passive dynamics

This system of $|\tilde{\mathcal{U}}(x)|$ equations has to be solved separately for each x , where $\tilde{p}, \tilde{\ell}$ are given and q, p are unknown. Let us fix x and define the vectors \mathbf{m}, \mathbf{b} and the matrix D with elements

$$\begin{aligned} m_{x'} &= \log p(x'|x) \\ b_a &= \tilde{\ell}(x, a) - \sum_{x'} \tilde{p}(x'|x, a) \log \tilde{p}(x'|x, a) \\ D_{ax'} &= \tilde{p}(x'|x, a) \end{aligned} \quad (37)$$

Then the above system of equations becomes linear:

$$q\mathbf{1} - D\mathbf{m} = \mathbf{b} \quad (38)$$

D, \mathbf{b} are given, q, \mathbf{m} are unknown, $\mathbf{1}$ is a column vector of 1's. In addition we require that p be a normalized probability distribution, which holds when

$$\sum_{x'} \exp(m_{x'}) = 1 \quad (39)$$

The latter equation is nonlinear but nevertheless the problem can be made linear. Since D is a stochastic matrix, we have $D\mathbf{1} = \mathbf{1}$. Then equation (38) is equivalent to

$$D(q\mathbf{1} - \mathbf{m}) = \mathbf{b} \quad (40)$$

unknown

which can be solved for $\mathbf{c} = q\mathbf{1} - \mathbf{m}$ using a linear solver. For any q the vector $\mathbf{m} = q\mathbf{1} - \mathbf{c}$ is a solution to (38). Thus we can choose q so as to make \mathbf{m} satisfy (39), namely

$$q = -\log \sum_{x'} \exp(-c_{x'}) \quad (41)$$

If D is row-rank-deficient the solution \mathbf{c} is not unique, and we should be able to exploit the freedom in choosing \mathbf{c} to improve the approximation of the traditional MDP. If D is column-rank-deficient then an exact embedding cannot be constructed. However this is unlikely to occur in practice because it essentially means that the number of symbolic actions is greater than the number of possible next states.

5 Car-on-a-hill simulation

The continuous control problem illustrated in **Fig. 5** in the main text is as follows. x_1 and x_2 denote the horizontal position and tangential velocity of the car. The state vector is $\mathbf{x} = [x_1, x_2]^T$. The hill elevation over position x_1 is

$s(x_1) = d \cdot x_2^2 / d \cdot x_1$. The larger x_1 is, the higher $s(x_1)$ becomes. The infimum is 0 when $x_1=0$ in the bottom

$$s(x_1) = 2 - 2 \exp(-x_1^2/2) \quad (42)$$

The slope is $s'(x_1) = 2x_1 \exp(-x_1^2/2)$ and the angle relative to the horizontal plane is $\text{atan}(s'(x_1))$. The tangential acceleration reflects the effects of gravity, damping, control signal u and noise. The dynamics are

$$\begin{aligned} dx_1 &= x_2 \cos(\text{atan}(s'(x_1))) dt \\ dx_2 &= -g \text{sgn}(x_1) \sin(\text{atan}(s'(x_1))) dt - \beta x_2 dt + u dt + d\omega \end{aligned} \quad (43)$$

$g = 9.8$ is the gravitational constant and $\beta = 0.5$ is the damping coefficient. The cost rate is $\ell(\mathbf{x}, u) = q + \frac{1}{2}u^2$ with $q = 5$. The goal states are all states such that $|x_1 - 2.5| < 0.05$ and

$|x_2| < 0.2$. This cost model encodes the task of parking at horizontal position 2.5 in minimal time and with minimal control energy. The constant q determines the relative importance of time and energy. Some error tolerance is needed because the dynamics are stochastic. This continuous problem is in the form given in the main text, so it can be approximated with an MDP in our class. It can also be approximated with a traditional MDP. Both approximations use the same state space discretization: a 101-by-101 grid spanning $x_1 \in [-3, +3]$, $x_2 \in [-9, +9]$. The traditional MDP also uses discretization of the control space: a 101 point grid spanning $u \in [-30, +30]$. The passive dynamics p are constructed by discretizing the time axis (with time step $h = 0.05$) and defining probabilistic transitions among discrete states so that the mean and variance of the continuous-state dynamics are preserved. The noise distribution is discretized at 9 points spanning ± 3 standard deviations in the x_2 direction, that is, $[-3\sqrt{h}, +3\sqrt{h}]$. The controlled dynamics are obtained from p by shifting in the x_2 direction. For each value of u the set of possible next states is a 2-by-9 sub-grid, except at the edges of the grid where non-existent states are removed and the distribution is normalized to sum to 1.

(a) – Schematic illustration of the problem.

(b) – Comparison of Z-iteration (blue), policy iteration (red) and value iteration (black). The vertical axis shows the empirical performance of the control policies. It was found by initializing 10 trajectories in each discrete state and sampling until the goal was reached or until the trajectory length exceeded 500 steps. The sampling was done in discrete time and continuous space, using the nearest discrete state to determine the control signal. The horizontal axis (note the log scale) shows the number of updates for each method. One update involves a computation of the form $A\mathbf{v} + \mathbf{b}$ for policy and value iteration, and $A\mathbf{z}$ for Z-iteration. The computation of minima in policy and value iteration is not counted, thus our method has an even bigger advantage than what is shown in the figure. The evaluation step in policy iteration was done with an iterative linear solver which was terminated at 20 iterations (or when convergence was reached) because complete evaluation slows down policy iteration. The value of 20 was manually optimized. Recall that the evaluation step in policy iteration, as well as the linear problem that needs to be solved in our formulation, can also be handled with a direct solver. Then our method becomes equivalent to a single evaluation step in policy iteration. Policy iteration using a direct solver converged in 10 iterations, thus our method was more than 10 times faster.

(c) – The optimal cost-to-go function computed by Z-iteration (blue is small; red is large). Also shown are two stochastic trajectories generated by the optimal controller (black). The magenta curve is the most likely trajectory of the optimally-controlled stochastic system. It is computed by solving the corresponding deterministic problem via dynamic programming applied to the discretization. Note that we could also solve a continuous deterministic problem (given in the main text) and recover the same trajectory.

(d) – The optimal cost-to-go $v(x)$ inferred from a dataset generated by the optimal controller. The dataset contained 20 state transitions per state: the system was initialized 20 times in each discrete state and the next discrete state was sampled from the optimal controller. The pixels shown in brown correspond to states where none of the transitions landed. The cost-to-go at those states cannot be inferred. The inference procedure is based on the diagonal Gauss-Newton method applied to the function $L(\mathbf{v})$ in the main text.