Stochasticity in Nonlinear Optimal Control

Nathan Ratliff

May 20, 2014

Abstract

Deterministic dynamics is a severe abstraction. We can never correctly predict precisely what's going to happen when we apply a given set of torques to a complex high dimensional robotic system, especially when the robot has significant inertia and is in contact with the unstructured environment. Assuming some level of stochasticity help us reason about these uncertainties and even choose actions to explicitly reduce uncertainty. This document studies stochasticity within nonlinear optimal control. We start by exploring to to properly model stochastic dynamics in a way that leads to action choices explicitly accounting for uncertainty. We derive a very general framework for reasoning about stochasticity over a long horizon in optimal control that allows us to leverage both general nonlinear optimization tools and common distributional approximations such as Expectation Propagation to ensure that the distribution over where the robot might be doesn't grow arbitrarily complex as it propagates along the trajectory. We then discuss how various modeling tools (entropy costs and Gaussian expectations) interaction with the optimization to produce explicit uncertainty reducing behaviors to better achieve desired goals.

1 The world's not deterministic

We can never model everything. Hoses for hydraulics hang from robots, what we thought were rigid body parts might bend in unspecified ways, our vision system might not be perfect, contacts are always nebulous and imprecise, and there may be something like wind in outdoor environments that we can't predict. Moreover, when we think we're applying a certain set of torques, that itself is just an abstraction. Really a lower level controller is tuning hydraulic valves or adjusting motor voltages in an attempt to achieve that desired torque on average at a really fast loop (often 1kHz).

The bottom line is, our models are never perfect. They're just mathematical niceties that *help* us predict how our system will behave. So, if our predicts are wrong every time we try to make a prediction, what do we do?

The good news is that the precise prediction might be wrong, but it might not be that wrong. Even if we can't predict *exactly* where the system will be

one time step in the future if we apply a particular set of torques to the motors, there's often some amount of regularity to the behavior that's relatively consistent in its distribution. Instead of being able to say that the robot will precisely be at some new state $\boldsymbol{s}_{t+1} = \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t)$ if we apply action $\boldsymbol{u}_t$ from state $\boldsymbol{s}_t$, we might instead be able to say that, although we don't know exactly where it is, we know that it's going to be in the general vicinity of some nonlinear function of our state and action $\boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t)$, distributed around that point as a Gaussian with covariance $\boldsymbol{C}(\boldsymbol{s}_t, \boldsymbol{u}_t)$, also a function of the state and action. This nonlinear Gaussian assumption (nonlinear because both the mean and covariance are nonlinear functions of the current state and action) is an explicit mathematical model of the uncertainty that plagues simpler mathematical models of dynamical systems.

Again, it's not perfect (we choose Gaussians, for instance, largely because their mathematical manipulation is simple), but it's better than nothing. More importantly, by explicitly modeling this stochasticity in the system, we can start asking questions about how that uncertainty affects the expected cost of a particular set of actions or how we might choose or optimize actions to explicitly reduce uncertainty.

This document explores how we can model uncertainty in dynamic systems and how those models play into optimizing behavior. We start with a discussion of how we might go about modeling stochasticity and then examine how stochasticity enter specifically into the problem of optimal control. This presentation diverges somewhat from classical approaches, which often characterize optimal behavior using the Hamilton Jacobi Bellman equation (there are a number of excellent references within that domain Todorov (2006)) in favor of a derivation that leads to a slightly more general framework that focuses on representing belief dynamics (how distributions over state transform from one time step to the next) explicitly in order to more easily leverage generic nonlinear optimization algorithms (such as Newton's method and Augmented Lagrangian Ratliff (2014)) and belief propagation approximations (such as Expectation Propagation Minka (2001); Toussaint (2009)). Later on we spend some time analyzing how specific formulations might induce behaviors that explicitly take action early on to reduce uncertainty to improve its overall performance in the end.

2 Modeling stochasticity

This section takes a closer look at what we require in a model of system stochasticity to make it useful for control. In particular, we see that the simples linear Gaussian model gives us no control at all over stochasticity and is therefore less interesting as a dynamic model of stochasticity for many purposes.

2.1 The uncontrollability of linear Gaussian noise

The simplest dynamical model we might consider is a stochastic generalization of a linear model. Linear models of the form $\boldsymbol{s}_{t+1} = \boldsymbol{A}_t \boldsymbol{s}_t + \boldsymbol{B}_t \boldsymbol{u}_t$, in conjunction

with quadratic costs, are very popular because their structure leads to easy mathematical manipulation and nice closed-form solutions for optimal control. Here, the next state is just a linear transformation of the current state plus a linear transformation of the action. If we have a quadratic cost function over states one step in the future $c(\boldsymbol{s}_{t+1})$, we can plug the linear model in to make a new quadratic $Q(\boldsymbol{s}_t, \boldsymbol{u}_t) = c(\boldsymbol{A}_t\boldsymbol{s}_t + \boldsymbol{B}_t\boldsymbol{u}_t)$, and then even optimize over actions $\boldsymbol{u}_t$ to produce a quadratic over states $V(\boldsymbol{s}_t) = \min_{\boldsymbol{u}_t} Q(\boldsymbol{s}_t, \boldsymbol{u}_t)$. Quadratic functions and linear systems beget quadratic functions making these models very convenient. But what happens now if we add stochasticity to the mix.

The traditional way of adding in some stochasticity is to add some Gaussian noise to the transition in the form of a linearly transformed zero-centered isometric random variable $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$. This linear-Gaussian system in full is

$$\boldsymbol{s}_{t+1} = \boldsymbol{A}_t\boldsymbol{s}_t + \boldsymbol{B}_t\boldsymbol{u}_t + \boldsymbol{S}_t\epsilon. \tag{1}$$

The transformation matrix $\boldsymbol{S}_t$ ensures that $\boldsymbol{S}_t\epsilon \sim \mathcal{N}(0, \boldsymbol{S}_t\boldsymbol{S}_t^T)$. Since this equation is just an affine transformation of the isometric Gaussian variable ϵ, the resulting distribution over next states also Gaussian with mean and covariance

$$\boldsymbol{s}_{t+1} \sim \mathcal{N}(\boldsymbol{A}_t\boldsymbol{s}_t + \boldsymbol{B}_t\boldsymbol{u}_t,\ \boldsymbol{S}_t\boldsymbol{S}_t^T). \tag{2}$$

So far so good. For a given state and action, this stochastic model turns the deterministic linear dynamics model into probability distributions centered around where the deterministic dynamics would have taken us but with an added covariance of $\boldsymbol{S}_t\boldsymbol{S}_t^T$ around that point. Moreover, iterating this equation just successively applies affine transformations to Gaussian random variables, so the final random variable will always be a Gaussian distribution of some variance centered around where the deterministic linear dynamics would have place the system.

But to make this stochasticity useful, we need to be able to link that stochasticity (in particular the covariance, in this case) back to the actions. We need to be able to choose actions that somehow reduce covariance or at least rotate the distribution to line up with some low cost region in a cost function. Bottom line: we need to be able to control the stochasticity. And here is where we run into trouble.

Lets examine the explicit distribution that results from iterating this linear-Gaussian stochastic model. Suppose we start from some distribution over states $\boldsymbol{s}_0 = \boldsymbol{\mu} + \boldsymbol{S}_0\epsilon$. Each application of Equation 1 multiplies the previous state random variable by $\boldsymbol{A}_t$, adds an offset $\boldsymbol{B}_t\boldsymbol{u}_t$ (deterministic), and adds some more Gaussian noise of the form $\boldsymbol{S}_t\epsilon$. What we care about currently is those Gaussian noise terms. In terms of those, each iteration simply transforms the previous noise term by $\boldsymbol{A}_t$ and adds a new one. So after T iterations, the resulting noise distribution will just be a sum of a bunch of terms of the form $\boldsymbol{A}_T \cdots \boldsymbol{A}_{k+1}\boldsymbol{S}_k\epsilon$, where k is the iteration where that term was introduced into the system. The rest of the terms (the non-noise terms) contribute only to the mean of the

Gaussian, which will ultimately land precisely where the deterministic linear system would place it $\boldsymbol{\mu}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_T)$. Thus, the final Gaussian distribution is

$$\boldsymbol{s}_{T+1} \sim \mathcal{N}\left(\boldsymbol{\mu}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_T),\ \boldsymbol{\Sigma}_{T+1} \right)$$
$$\text{where} \quad \boldsymbol{\Sigma}_{T+1} = \left(\sum_{t=0}^{T} \boldsymbol{A}_T \cdots \boldsymbol{A}_{t+1} \boldsymbol{S}_t \right) \left(\sum_{t=0}^{T} \boldsymbol{A}_T \cdots \boldsymbol{A}_{t+1} \boldsymbol{S}_t \right)^T .$$

What's striking about this equation is that, although we can control the mean of this distribution (it's the same function of actions as the deterministic version of the system), but the covariance at time $T+1$ is entirely independent of the actions. It changes with each time step as a function of the linear dynamics transformation $\boldsymbol{A}_t$ and the added noise $\boldsymbol{S}_t \epsilon$, but no matter what actions we choose, it will always be the same after a given number of time steps. We can shift around where the the Gaussian distribution at time step $T+1$ is centered, but we can't change the shape of that distribution for a given time.

That restriction is severe. It means, no matter what we do, we can't choose actions early on to reduce uncertainty to improve performance later. Indeed, if all of our cost functions were quadratic (specifically, symmetric around the principle axes and uni-modal) we can simply optimize the system using the deterministic dynamics and then calculate what the covariance would around each time step to find the optimal controller. We'd like a model that's more expressive than that.

2.2 An example of what we need: Nonlinear Gaussian dynamics

The previous section demonstrated that linear Gaussian dynamics aren't enough to control the stochasticity in interesting ways. This section gives an example of a common model that does.

Consider a more general nonlinear Gaussian model of the form

$$\boldsymbol{s}_{t+1} = \boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t) + \boldsymbol{F}(\boldsymbol{s}_t, \boldsymbol{u}_t)\epsilon, \tag{3}$$

where, again, $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$. The conditional dynamics (probability of the next state *given* the current state and action) are still Gaussian:

$$p(\boldsymbol{s}_{t+1} | \boldsymbol{s}_t, \boldsymbol{u}_t) \sim \mathcal{N}\left(\boldsymbol{f}(\boldsymbol{s}_t, \boldsymbol{u}_t),\ \boldsymbol{F}(\boldsymbol{s}_t, \boldsymbol{u}_t)\boldsymbol{F}(\boldsymbol{s}_t, \boldsymbol{u}_t)^T \right). \tag{4}$$

However, that Gaussian's mean and covariance are both *nonlinear* functions of the state and action. That nonlinearity, particularly as a function of the action $\boldsymbol{u}_t$, means that we can control both where that mean is at time step T as well as how the covariance is shaped there.

This is just one such model that fits our needs. In particular it results in the Markov Belief Dynamics discussed in Section 5.

3 Optimizing expected performance

Given a stochastic model of dynamics, any sequence of T actions $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_T$ produces not only a single distinct state $\boldsymbol{s}_{T+1}$, but an entire distribution over all states we've seen up to that point $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{T+1} | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T)$. A reasonable generalization of performance cost is then simply the expected value of the type of cost function we may have used before for deterministic systems: we want our expected performance to be good. Explicitly, the resulting optimization problem would be

$$\min_{\boldsymbol{u}_{1:T}} \quad E_{\boldsymbol{s}_{1:T+1} | \boldsymbol{u}_{1:T}} \left[\sum_{t=1}^{T} c_t(\boldsymbol{s}_t, \boldsymbol{u}_t) + \psi(\boldsymbol{s}_{T+1}) \right]. \tag{5}$$

Since the original objective was a sum over terms, the expected value, which itself is an integral over all the state variables, decomposes in a very intuitive way across the terms:

$$\begin{aligned}
&\int d\boldsymbol{s}_1 \cdots d\boldsymbol{s}_{T+1} \; p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{T+1} | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) \; \left(\sum_{t=1}^{T} c_t(\boldsymbol{s}_t, \boldsymbol{u}_t) + \psi(\boldsymbol{s}_{T+1}) \right) \qquad (6) \\
&= \sum_{t=1}^{T} \int d\boldsymbol{s}_1 \cdots d\boldsymbol{s}_{T+1} \; p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{T+1} | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) \; c_t(\boldsymbol{s}_t, \boldsymbol{u}_t) \\
&\qquad\qquad + \int d\boldsymbol{s}_1 \cdots d\boldsymbol{s}_{T+1} \; p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{T+1} | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) \; \psi(\boldsymbol{s}_{T+1}) \\
&= \sum_{t=1}^{T} \int d\boldsymbol{s}_t \; p(\boldsymbol{s}_t | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1}) \; c_t(\boldsymbol{s}_t, \boldsymbol{u}_t) + \int d\boldsymbol{s}_{T+1} \; p(\boldsymbol{s}_{T+1} | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) \; \psi(\boldsymbol{s}_{T+1}) \\
&= \sum_{t=1}^{T} E_{\boldsymbol{s}_t | \boldsymbol{u}_{1:t-1}} \left[\; c_t(\boldsymbol{s}_t, \boldsymbol{u}_t) \; \right] \; + \; E_{\boldsymbol{s}_{T+1} | \boldsymbol{u}_{1:T}} \left[\; \psi(\boldsymbol{s}_{T+1}) \; \right].
\end{aligned}$$

For that second-to-last step, we used the property $p(\boldsymbol{s}_t | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) = p(\boldsymbol{s}_t | \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1})$, which is simply a statement that, in the dynamics functions we're considering, the marginal distribution over states at time t depends only on past actions, not future actions. In other words, the full expected cost is equivalently understood as a sum of individual marginal expectations. We never have to fully represent that joint distribution over all states given the prescribed action sequence. All we need to know is the local distribution over states at *any moment in time* given the actions we've taken to get there. Adding up individual expected per-time-slice costs constructs a general objective that measures the global expectation.

4 A generalization: belief dynamics

Section 3 above decomposed a classic setting in which the goal was to optimize expected long term performance, resulting in an objective that depended on the

per-time-slice marginal distributions $p(\boldsymbol{s}_t|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1})$ over where the robot might be at time t given that it's taken actions $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1}$ (previously) to get there. This section explores those marginals in more detail, which we can think of as our *belief* over where the robot might be at time t given a set of actions. We, in particular, examine various constructions of *belief dynamics* which formalize how those belief distributions changes over time under specific modeling assumptions.

5 Markov belief dynamics

The most common form of belief dynamics is known as Markov belief dynamics. Markov belief dynamics make both the common assumption that the current state depends only on past actions, but also the more strict assumption that knowing the state now tells us everything we need to know about where the robot might be in the future given a set of future actions. Mathematically, that says

$$p(\boldsymbol{s}_{t+1}, \ldots, \boldsymbol{s}_{T+1}|\boldsymbol{s}_1, \ldots, \boldsymbol{s}_t, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_T) = p(\boldsymbol{s}_{t+1}, \ldots, \boldsymbol{s}_{T+1}|\boldsymbol{s}_t, \boldsymbol{u}_t, \ldots, \boldsymbol{u}_T). \tag{7}$$

Consider now the marginal distribution $p(\boldsymbol{s}_{t+1}|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t)$ over states one time step in the future given some past actions. This expression is an integral over the joint distribution over $\boldsymbol{s}_t$ and $\boldsymbol{s}_{t+1}$, and that integral decomposes in a nice way because of the Markov assumption of Equation 7:

$$\begin{aligned} p(\boldsymbol{s}_{t+1}|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t) &= \int d\boldsymbol{s}_t \; p(\boldsymbol{s}_{t+1}, \boldsymbol{s}_t|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t) \\ &= \int d\boldsymbol{s}_t \; p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_t) \; p(\boldsymbol{s}_t|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_t) \\ &= \int d\boldsymbol{s}_t \; p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{u}_t) \; p(\boldsymbol{s}_t|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1}). \end{aligned}$$

The second-to-last step here used the chain-rule for probabilities to write the joint distribution as a conditional times a marginal and the last step invoked the Markov assumption in Equation 7.

This relationship tells us that if we have a distribution $p_t(\boldsymbol{s}_t) = p(\boldsymbol{s}_t|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1})$ over states at time t, the *dynamics* of how it transforms (under the Markov assumption) into a distribution over states at the next time step is given by

$$p_{t+1}(\boldsymbol{s}_{t+1}) = \int d\boldsymbol{s}_t \; p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{u}_t) \; p_t(\boldsymbol{s}_t). \tag{8}$$

These belief dynamics are entirely governed by the action transition probabilities $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{u}_t)$ that describe where we might end up if we executed action $\boldsymbol{u}_t$ from state $\boldsymbol{s}_t$. The rest of this equation then derives simply from the rules of probability. The next section, generalizes this idea to allow for arbitrary transformations of the probability distributions, which may include approximations to reduce computational requirements.

5.1 Generalized belief dynamics

More abstractly, the expression in Equation 8 suggests that what we really care about is the dynamics of belief transitions: how does the distribution over where the robot is now change over one time step under the influence of a given action. Specifically, representing the belief distribution $p_t(\boldsymbol{s}_t)$ using a *belief state* variable $\boldsymbol{b}_t$ (composed of whatever it takes to parametrize the distribution, e.g. the distribution's sufficient statistics), the question of how the belief distribution transforms from one time step to the next under a given action can be represented using a function that transforms the belief state $\boldsymbol{b}_t$ to a new belief state $\boldsymbol{b}_{t+1}$. Specifically, we can represent these belief state dynamics as

$$\boldsymbol{b}_{t+1} = \boldsymbol{f}(\boldsymbol{b}_t, \boldsymbol{u}_t). \tag{9}$$

Writing the transition this way emphasizes the relationship between this belief state dynamics function and the non-stochastic dynamics function modeling only deterministic state-transitions.

As a concrete example, each $\boldsymbol{b}_t$ may be the sufficient statistics of a Gaussian distribution, which are just the distribution's mean and covariance, and $\boldsymbol{b}_{t+1} = \boldsymbol{f}(\boldsymbol{b}_t, \boldsymbol{u}_t)$ may represent how the mean and covariance at time t transform to a new mean and covariance at the next time step $t+1$ under action $\boldsymbol{u}_t$.

Importantly, the belief dynamics represented by Equation 9 may be precisely the Markov dynamics as established in Equation 8. But they don't have to be. Whatever makes sense for a particular problem and helps develop uncertainty cognizant behavior is fair game. Indeed, typically the belief representation in the most general case grows with every time step. In the above Gaussian transition example, such a belief state transition can only be exact when the dynamics are linear Gaussian as given in Equation 1. In general, the true Markov belief dynamics of a nonlinear system will turn a Gaussian distribution into a potentially arbitrarily complex non-Gaussian, even multi-modal, distribution which becomes increasingly hard to represent.

A common approximation for these cases, is to use a technique called Expectation Propagation (EP) Minka (2001). The EP approximation projects the resulting posterior distribution back onto a fixed family of distributions, such as Gaussian distributions to ensure the dynamics remain tractable with a consistent representation size over time. In that case, both $\boldsymbol{b}_t$ and $\boldsymbol{b}_{t+1}$ may contain the mean and covariance of Gaussian distributions. $\boldsymbol{f}$, then, may calculate a nonlinear transformation of the Gaussian (by truncating it, for instance, to have support only on one side of a constraint surface) followed by a projection back onto the space of Gaussian distributions by finding the Gaussian distribution that minimizes the KL-divergence between the true posterior and the family of Gaussians. Such EP truncated Gaussian approximations are described in detail in Toussaint (2009) where they're used to model the interaction of the distributions with hard constraints in the environment such as walls or other surfaces.

5.2 Distributional costs

Finally, this formulation gives us more control on how we define the objective. We're free to use the information in each belief state $\boldsymbol{b}_t$ to define objective terms as expected per-time-slice state-action costs (corresponding to the classical expected cost-to-go formulation), but we can do more. If desired we can add heuristic cost terms that are explicitly a function of the belief distribution itself that don't necessarily implement expectations. Explicitly, the resulting generalized constrained optimization problem becomes

$$\begin{aligned} \min_{\boldsymbol{u}_{1:T}} & \sum_{t=1}^{T} c_t(\boldsymbol{b}_t, \boldsymbol{u}_t) + \psi(\boldsymbol{b}_{T+1}) \\ \text{s.t.} \quad & \boldsymbol{b}_{t+1} = \boldsymbol{f}(\boldsymbol{b}_t, \boldsymbol{u}_t) \\ & \boldsymbol{g}_t(\boldsymbol{b}_t) \geq 0 \qquad \forall t \\ & \boldsymbol{h}_t(\boldsymbol{b}_t) = 0 \qquad \forall t. \end{aligned} \tag{10}$$

Note that this formulation is exactly the same as the deterministic optimal control formulation Ratliff (2014), except now we have *belief state* variables $\boldsymbol{b}_t$ and *belief state* dynamics. Importantly, under this formulation, it becomes clear that we can use many of the same optimization methods described in Ratliff (2014) for Optimal Control to optimize Stochastic Optimal Control problems, too, as long as we have a tractable model of the belief dynamics given by $\boldsymbol{f}$.

The next section gives an example of how, under this formulation, specifically adding belief state entropy penalization functions can induce uncertainty reducing behaviors. And then Section 5.2.2 discusses the intuition behind how even simple terminal state expectations can automatically induce variance reduction actions earlier on as part of the optimal solution.

5.2.1 Entropy heuristics

Classically, Stochastic Optimal Control often optimizes an expected-cost formulation such as that given in Equation 5. Equations 6 demonstrate that that we may also view this global expectation over cumulative costs as a sum of individual expected per-time-slice cost terms. That means, at each time step, the belief distribution $\boldsymbol{p}_t(\boldsymbol{s}_t|\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{t-1})$ only enters into the cost function through an expectation operator of the form

$$E_{\boldsymbol{s}_t|\boldsymbol{u}_{1:t-1}}[\, c_t(\boldsymbol{s}_t, \boldsymbol{u}_t)\,]. \tag{11}$$

In particular, we only design cost functions over states and never over the belief distribution itself.

But the general formulation of Equation 10 offers a more flexible modeling interface. Under this formulation, general cost terms $c_t(\boldsymbol{b}_t, \boldsymbol{u}_t)$ (or terminal costs $\psi(\boldsymbol{b}_{T+1})$) are explicitly over the belief state and they can be arbitrary as long as they're differentiable.

Suppose, for instance, we heuristically believe that broad (high uncertainty) belief distributions are bad for the task at hand (e.g. precise manipulation). We can define a cost term that explicitly penalizes the entropy[1] of the belief $\mathcal{H}(\boldsymbol{b}_t)$ at each iteration in addition to our expected cost $E_{\boldsymbol{b}_t}[\widetilde{c}_t(\boldsymbol{s}_t, \boldsymbol{u}_t)]$ for some state-action cost $\widetilde{c}_t$. Our resulting intermediate cost terms will then be

$$c_t(\boldsymbol{b}_t, \boldsymbol{u}_t) = E_{\boldsymbol{b}_t}[\widetilde{c}_t(\boldsymbol{s}_t, \boldsymbol{u}_t)] + \alpha \mathcal{H}(\boldsymbol{b}_t), \tag{12}$$

where $\alpha > 0$ is some positive trade-off parameter. This cost term will attempt to minimize expected cost as before, but also explicitly attempt to choose actions earlier on that decrease the entropy of the belief. If, for instance, our belief dynamics function models the uncertainty reduction inherent in moving closer to a wall (since we know we aren't punching through the wall), these entropy terms will induce a biased behavior that favors staying closer to surfaces so that the robot can be more certain of where it is at any moment in time.

5.2.2 Expected terminal costs and covariance pressure

This section discusses how expected terminal costs, themselves, can put pressure on the covariance of the final belief $\boldsymbol{b}_{T+1}$ which, itself, can result in behaviors similar to those described in the previous section that favor actions earlier on that explicitly attempt to reduce belief uncertainty.

Consider a quadratic state-terminal cost of the form

$$\psi(\boldsymbol{s}_{T+1}) = \frac{1}{2}\|\boldsymbol{s}_{T+1} - \boldsymbol{s}_d\|^2, \tag{13}$$

and suppose we use belief dynamics that project each nonlinear transition back onto the space of Gaussian distributions at each iteration. In other words, although the true Markov transition might result in a non-Gaussian distribution our belief dynamics will subsequently approximate that true resulting distribution with a Gaussian (for instance, using the Expectation Propagation approximation) so at every time step, $\boldsymbol{b}_t$ will always be Gaussian.

Since $\boldsymbol{b}_{T+1} \sim \mathcal{N}(\boldsymbol{\mu}_{T+1}, \boldsymbol{\Sigma}_{T+1})$ is a Gaussian, the expectation $E_{\boldsymbol{b}_{T+1}}[\psi(\boldsymbol{s}_{T+1})]$ is a Gaussian expectation of a quadratic, which we can calculate analytically Toussaint (2009) as

$$E_{\boldsymbol{b}_{T+1}}\left[\frac{1}{2}\|\boldsymbol{s}_{T+1} - \boldsymbol{s}_d\|^2\right] = \frac{1}{2}\|\mu_{T+1} - \boldsymbol{s}_d\|^2 + \underbrace{\mathbf{trace}\,(\boldsymbol{\Sigma}_{T+1})}_{\sum_{i=1}^{d}\sigma_i}, \tag{14}$$

where $\{\sigma_i\}_{i=1}^{d}$ are the d Eigenvalues of the covariance matrix $\boldsymbol{\Sigma}_{T+1}$. This trace term reflects the fact that it not only matters that the mean is at the right spot, but we also need the covariance to be as tight around that point as possible. This trace term explicitly penalizes broad covariance, and is actually qualitatively

[1]The entropy $\mathcal{H}(\boldsymbol{b}_t)$ of the distribution represented by belief state $\boldsymbol{b}_t$ is a measure of uncertainty taken from Information Theory Cover & Thomas (2006).

similar to the entropy of the Gaussian distribution. The differential entropy of a Gaussian distribution $p(\boldsymbol{x}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is

$$\mathcal{H} = c + \frac{1}{2} \log |\boldsymbol{\Sigma}| = c + \sum_{i=1}^{d} \log \sigma_i,$$

where c is a constant, $|\boldsymbol{\Sigma}|$ is the determininant of $\boldsymbol{\Sigma}$, and σ_i^2 are the Eigenvalues of $\boldsymbol{\Sigma}$. (The Eigenvalues are variances along the principle axes (Eigenvectors), and their square roots σ_i are the corresponding standard deviations along those axes.)

In the same way that the entropy terms we introduced in Section 5.2.1 biased the optimal controller toward actions earlier on that attempted to reduce uncertainty in the distribution, this expectation, itself, as the terminal cost, creates a *pressure* on the terminal covariance that propagates back through the trajectory to coax earlier actions toward solutions that reduce uncertainty.

References

Cover, Thomas M. and Thomas, Joy A. *Elements of information theory*. Wiley-Interscience, 2nd edition edition, 2006.

Minka, Thomas. Expectation propagation for approximate bayesian inference. In *Proceeds of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2001.

Ratliff, Nathan. Nonlinear optimal control: Reductions to newton optimization, 2014. Lecture notes.

Todorov, E. Optimal control theory. *In Bayesian Brain: Probabilistic Approaches to Neural Coding*, pp. 269–298, 2006.

Toussaint, Marc. Pros and cons of truncated gaussian ep in the context of approximate inference control. In *NIPS Workshop on Probabilistic Approaches for Robotics and Control*, 2009.