Primitives for optimal control

Emanuel Todorov, UCSD

The sensorimotor system finds near-optimal solutions to control problems too hard to handle with existing algorithms, and does so in real time. How is this possible? One hypothesis is that movements are constructed from primitives which somehow simplify control. While the implied reduction in dimensionality is well documented, the structure and origin of the hypothetical primitives, the rules for combining them and the way in which they simplify control remain unclear. Even less clear is how to reconcile the powerful ideas of compositionality and optimality.

Here we report a mathematical breakthrough: we show how optimal feedback control laws for a wide range of tasks can be composed from certain task-independent primitives. These primitives are eigenfunctions of the stochastic plant dynamics, and can be learned through random exploration in a model-free way. Key to our results is the discovery of a class of nonlinear control problems whose optimal solutions satisfy a linear, and thus decomposable, partial differential equation.

Our results apply to stochastic optimal control problems in the general form

$$\begin{aligned} &\text{plant dynamics:} \quad d\mathbf{x}(t) = \mathbf{a}(\mathbf{x}(t))\,dt + B(\mathbf{x}(t))\,(\mathbf{u}(t)\,dt + \sigma d\omega(t)) \\ &\text{cumulative cost:} \quad h(\mathbf{x}(t_f)) + \int_0^{t_f} \frac{r}{2}\,\|\mathbf{u}(t)\|^2\,dt \end{aligned} \tag{1}$$

where $h, \mathbf{a}, B$ are arbitrary differentiable functions. Many biomechanical control problems can be cast in this form. The only unusual assumption here is that the noise and control act in the same subspace. The optimal controls are easily found given the gradient $v_{\mathbf{x}}$ of the optimal value function:

$$\text{optimal feedback control law:} \quad \mathbf{u}^*(\mathbf{x},t) = -r^{-1}B(\mathbf{x})^{\mathsf{T}}\,v_{\mathbf{x}}(\mathbf{x},t) \tag{2}$$

$v(\mathbf{x},t)$ is the cost expected to accumulate if we start at state $\mathbf{x}$ and time t and act optimally until the final time t_f; $v(\mathbf{x},t_f) = h(\mathbf{x})$. It satisfies the Hamilton-Jacobi-Bellman (HJB) equation

$$\text{HJB equation:} \quad -v_t = \mathbf{a}^{\mathsf{T}}v_{\mathbf{x}} + \frac{\sigma^2}{2}\,\text{trace}\left(BB^{\mathsf{T}}v_{\mathbf{xx}}\right) - \frac{r^{-1}}{2}v_{\mathbf{x}}^{\mathsf{T}}BB^{\mathsf{T}}v_{\mathbf{x}} \tag{3}$$

This equation is nonlinear due to the last term. Nonlinearity is a major obstacle along the path to compositionality. However we have found a way to remove it – via a nonlinear "change of coordinates" which makes (3) exactly linear. Defining the function

$$\text{transformed optimal value function:} \quad z(\mathbf{x},t) = \exp\left(-r^{-1}\sigma^{-2}v(\mathbf{x},t)\right) \tag{4}$$

we express v and its derivatives in terms of z and its derivatives, substitute in (3), and obtain

$$r\sigma^2\frac{z_t}{z} = -r\sigma^2\mathbf{a}^{\mathsf{T}}\frac{z_{\mathbf{x}}}{z} - \frac{r\sigma^4}{2}\,\text{trace}\left(BB^{\mathsf{T}}\left(\frac{z_{\mathbf{xx}}}{z} - \frac{z_{\mathbf{x}}z_{\mathbf{x}}^{\mathsf{T}}}{z^2}\right)\right) - \frac{r\sigma^4}{2}\frac{z_{\mathbf{x}}^{\mathsf{T}}}{z}BB^{\mathsf{T}}\frac{z_{\mathbf{x}}}{z} \tag{5}$$

Multiplying both sides by $r^{-1}\sigma^{-2}z \neq 0$ and simplifying yields the following equation in terms of z:

$$\text{transformed HJB equation:} \quad -z_t = \mathbf{a}^{\mathsf{T}}z_{\mathbf{x}} + \frac{\sigma^2}{2}\,\text{trace}\left(BB^{\mathsf{T}}z_{\mathbf{xx}}\right) \tag{6}$$

This is identical to the HJB equation (3) except that the nonlinear term has vanished. The simplification occurred because our transformation broke down the noise contribution into two terms, one cancelling the nonlinear term exactly and the other being linear. Thus the noise is crucial.

The HJB equation is the foundation of stochastic optimal control theory, and making it linear has far-reaching implications not only for biological motor control but for control in general. Such linearity simplifies the search for solutions in general, and enables compositionality in particular.

Equation (6) can be written as $-z_t = \mathcal{L}[z]$ with the help of the linear differential operator

$$\mathcal{L}[f(\cdot)](\mathbf{x}) = \mathbf{a}(\mathbf{x})^{\mathsf{T}} f_{\mathbf{x}}(\mathbf{x}) + \frac{\sigma^2}{2} \operatorname{trace}\left(B(\mathbf{x}) B(\mathbf{x})^{\mathsf{T}} f_{\mathbf{xx}}(\mathbf{x})\right) \tag{7}$$

Let λ_i and $f^i(\cdot)$ be eigenvalues and eigenfunctions of $\mathcal{L}$, meaning that for all $\mathbf{x}$ they satisfy

$$\text{eigenvalue problem:} \quad 0 = \lambda_i f^i(\mathbf{x}) + \mathcal{L}\left[f^i(\cdot)\right](\mathbf{x}) \tag{8}$$

The idea is to use these eigenfunctions as primitives (or bases) for decomposing the function z:

$$\text{basis function decomposition:} \quad z(\mathbf{x}, t) = \textstyle\sum_i b_i(t) f^i(\mathbf{x}) \tag{9}$$

Differential operators such as $\mathcal{L}$ have infinitely many orthogonal eigenfunctions, which form a complete basis for the space of piece-wise continuous functions. For example, $\mathcal{L}[f] = \operatorname{trace}(f_{\mathbf{xx}})$ corresponds to Laplace's equation whose eigenfunctions are the harmonic functions. In practice the sum in (9) will be finite, resembling a truncated Fourier series.

We have now reduced the optimal control problem (1) to computing the time-varying scalar coefficients $b_i(t)$. Combining (9, 8, 7, 6) we see that these coefficients can be found analytically:

$$\frac{d}{dt} b_i(t) = \lambda_i b_i(t), \qquad b_i(t) = \exp(\lambda_i (t - t_f)) b_i(t_f) \tag{10}$$

once the boundary values $b_i(t_f)$ are known. For orthonormal $f^i(\cdot)$ the boundary values are found by simply projecting the transformed final cost onto the eigenfunctions:

$$b_i(t_f) = \int z(\mathbf{x}, t_f) f^i(\mathbf{x}) \, d\mathbf{x} = \int \exp\left(-r^{-1}\sigma^{-2} h(\mathbf{x})\right) f^i(\mathbf{x}) \, d\mathbf{x} \tag{11}$$

The optimal control law (2) is then a state-dependent linear combination of gradient vector fields:

$$\text{modular control law:} \quad \mathbf{u}^*(\mathbf{x}, t) = \sigma^2 B(\mathbf{x})^{\mathsf{T}} \textstyle\sum_i \dfrac{b_i(t)}{z(\mathbf{x}, t)} f^i_{\mathbf{x}}(\mathbf{x}) \tag{12}$$

The remaining question is how to find eigenfunctions $f^i(\cdot)$ satisfying (8), and in particular how the sensorimotor system might do that through learning. Such learning turns out to be possible using a model-free algorithm, which only needs access to state sequences sampled from the uncontrolled stochastic dynamics $d\mathbf{x} = \mathbf{a}(\mathbf{x}) dt + \sigma B(\mathbf{x}) d\omega$. The algorithm exploits the fact the *generator* of the above stochastic process happens to be $\mathcal{L}$, meaning that

$$\mathcal{L}[f(\cdot)](\mathbf{x}(t)) = \lim_{\Delta \to 0} E\left[\frac{f(\mathbf{x}(t + \Delta)) - f(\mathbf{x}(t))}{\Delta}\right] \tag{13}$$

In other words, $\mathcal{L}$ is the expected directional derivative along uncontrolled state trajectories. This yields an equivalent but more useful definition of an eigenfunction: a function which at every state is proportional to its expected rate of change under the uncontrolled dynamics. Now let $g(\mathbf{x}; \mathbf{w})$ be a function approximator (say a neural network with one output unit) with parameters $\mathbf{w}$. To make g approximate an eigenfunction of $\mathcal{L}$, we need to adjust the parameters $\mathbf{w}$ and the eigenvalue λ so as to minimize the error function

$$E(\mathbf{w}, \lambda) = \sum\nolimits_n \left(\frac{g(\mathbf{x}_{n+1}; \mathbf{w}) - g(\mathbf{x}_n; \mathbf{w})}{\Delta} + \lambda g(\mathbf{x}_n; \mathbf{w})\right)^2 \tag{14}$$

where $\mathbf{x}_1, \mathbf{x}_2, \cdots$ is a state trajectory sampled in discrete time with time-step Δ. Such minimization can be performed via gradient descent and will converge to one of the eigenpairs of $\mathcal{L}$, modulo local minima of course. To find several eigenpairs simultaneously we need to train multiple function approximators (say a neural network with multiple output units) on the same error function, but with an additional term which forces the outputs to be orthonormal. Methods for enforcing orthonormal outputs have already been developed in the neural network literature in the context of learning principal components (Oja's and Sanger's rule), and can be adapted to the present problem.