Latest News

Tom Westerdale

September 2, 2024

We have finally managed to show that strings of actions in learning systems and strings of genes in evolving systems change their probabilities according to the same simple formula, at least for a large class of learning adaptive plans. This is of course what John Holland implied when he maintained that learning and evolution are similar adaptive processes [1], but it is only now that we have been able to show that the equations of probability change are indeed similar for learning and evolution. Learning and evolution have much in common

Strings of actions are strings in time, whereas strings of genes are strings in space. The probabilities of action strings change as the system learns, just as the probabilities of gene strings change as a population evolves. The action strings can be thought of as constituting a virtual population in much the same way that gene strings constitute a real evolving population. A string's probability is its prevalence in the population, a virtual population in the case of learning, a real population in the case of evolution.

In both learning and evolution, adaptation tries to increase the probabilities of the more valuable strings. A string's value is a measure of the payoff we get if we use that string.[1] But there seemed to be a difference between evolution and learning. Gene string probabilities were unconditional probabilities, but action probabilities were conditional given the current state or given the previous actions. This difference is easiest to see when the learning system is a finite state Markov chain[2] in which each state has a payoff number. That number is the payoff you get every time you enter that state. An action is simply a state transition. The probability P_{ij} of transition $i \to j$ is the *conditional* probability that the next state is j given that the current state is i .

Learning causes a change in the conditional transition probabilities P_{ij} . But evolution causes changes in the unconditional gene probabilities. The learning analogue is the *unconditional* probability F_{ij} of transition $i \to j$. To compare learning with evolution we need to know how learning changes the unconditional probabilities F_{ij} , which I call the transition *frequencies.*

The transition probabilities do determine the frequencies, but in a complicated way. We have $F_{ij} = \tilde{p}_i P_{ij}$, where $\tilde{p}_i$ is the unconditional probability of being in state i . But the state probabilities $\tilde{p}_i$ are complicated functions of the transition probabilities P_{ij} . The learning plan specifies the rate of change P'_{ij} of the conditional probabilities, but to compare with evolution we need the rate of change F'_{ij} of the frequencies. How can we get a usable formula for F'_{ij} ? There is a trick, which we finally discovered, and it is the purpose of this note to explain the trick.

In learning, adaptive plans are typically based on state values, either explicitly or implicitly. The trick involves looking carefully at what we mean by value. Let m_i be the payoff attached to state i . Let $\bar{m}$ be the average payoff we get per time step. Learning tries to change the probabilities in a way that increases $\bar{m}$. I use the phrase *excess payoff* to mean payoff minus $\bar{m}$. We define

$a_i = m_i - \bar{m}$,

so a_i is the excess payoff of state i . It might very well be negative.

The *post-value* c_i of state i is the expectation of the sum of all the excess payoffs you would get on and after a visit to that state.[3] Usually post-value is simply called value. I define the *choice value* h_{ij} of transition $i \to j$ as follows.

$h_{ij} = c_j - c_i + a_i$.

It tells you how much better transition $i \to j$ is than the average transition from i .

[1] In Reinforcement Learning literature, the word "reward" means payoff, whereas in Evolutionary Computation literature, "reward" usually means reinforcement. In the literature, the word "fitness" sometimes means value, but it sometimes means reproductive rate.

[2] The chain should be strongly connected, so the transition probabilities completely determine the state probabilities.

[3] It's an infinite sum whose n'th term is the expectation of the excess payoff you would get n time steps after the visit. Provided the Markov chain is strongly connected, the sum converges in the Cesaro sense.

There is a large class of learning plans in which the rate of change P'_{ij} is given by
$P'_{ij} \; = \; KP_{ij}h_{ij}$.
The number K is the learning rate constant. I call these plans *natural plans*.[4]

You might think it would be simpler to simply specifiy $P'_{ij} \; = \; KP_{ij}c_j$, but that would send the P_{ij} numbers off to where the probabilities no longer add up to one. You would have to keep normalizing the probabilities, and that would result in a natural plan $P'_{ij} \; = \; KP_{ij}h_{ij}$. So there are lots of different scenarios that give you a natural plan. I won't talk about them here because I want to ask the crucial question. We know that $P'_{ij} \; = \; KP_{ij}h_{ij}$, but what is F'_{ij} ?

Of course the frequency derivatives F'_{ij} are always a function of the transition probability derivatives P'_{ij} , and we can use that function to obtain a formula for F'_{ij} in a natural plan, but that's a terribly complicated formula. How do we obtain a useful formula for F'_{ij} in a natural plan?

Now here is the trick. We ask the question, "What if time ran the other way?" A visit to a state is correlated with payoff received both before and after the visit. But the post-value looks only at the payoff after. We are missing half of the payoff.

Let B_{ij} be the conditional probability that the *previous* state was j given that the current state is i . The probabilities B_{ij} are the transition probabilities in a different Markov chain, a chain in which time runs in the other direction. I call it the *backward chain*, and it's often called the time reversed chain in the literature.

We look at the backward chain and ask what the post-value b_i of state i is in the backward chain. That's the *pre-value* of i in the original forward chain. The definition of b_i is just like c_i except that it uses the probabilities B_{ij} instead of P_{ij} . The pre-value b_i is the payoff on and *before* the visit to state i . It's the missing half of the payoff.

We can now define the *total value* v_{ij} of transition $i \to j$.
$v_{ij} \; = \; b_i + c_j$.
And now we can give our formula for F'_{ij} in a natural plan.
$F'_{ij} \; = \; KF_{ij}v_{ij}$.
It's that simple. We have the following theorem.

Theorem 1
$P'_{ij} \; = \; KP_{ij}h_{ij}$ *for all* ij
if and only if
$F'_{ij} \; = \; KF_{ij}v_{ij}$ *for all* ij .

In retrospect it looks obvious, but the proof is a bit tricky. The backward implication is easy, but turning the backward implication around to obtain the forward implication involves establishing a relation between the P' matrices and the F' matrices and then showing the relation is one to one. For all the details see [2].

If σ is the transition string $i \to j \to k \to \ell$, then its frequency is $F_\sigma \; = \; \tilde{p}_i P_{ij} P_{jk} P_{k\ell}$, and its total value is $v_\sigma \; = \; b_i + a_j + a_k + c_\ell$. We can similarly define the total value v_σ of any transition string σ , and in a natural plan we have
$F'_\sigma \; = \; KF_\sigma v_\sigma$ for all strings σ ,
just as in an evolving population of gene strings. The equation holds for all strings σ of any length and for all their substrings. The virtual population of transition strings has the same hierarchical structure as an evolving population of organisms. All the details and proofs are given in [2].

That's fine for a Markov chain, but what if the Markov property doesn't hold? I'll now briefly mention how we have extended theorem 1 to familiar rule based systems and to a simple actor-critic system. In these systems an action stands for a whole amalgamation of transitions. We look at systems that use a temporal difference method to estimate the values of the various actions, of the various amalgamations. The system adapts as if the post-value of each transition were the estimated value of the amalgamation it is in, so the transition post-values it uses are wrong. They are not even approximations of the true post-values in any normal sense of the word. I call them false post-values.

Temporal difference methods are *excellent* methods for reasons we won't discuss here, so it's important to understand the false post-values. Let me outline a trick we use to analyse them. It turns out that the true post-values are a simple invertible linear function of the excess payoffs, so the excess payoffs are a simple linear function of the true post-values. We apply that function to the false post-values to obtain what I call the false payoffs. Using these, we define the false pre-values and false total values. Then theorem 1 holds with all the true values replaced by the false values. The system is adapting and evolving like a

[4] In a natural plan we have $\bar{m}' \; = \; K \sum_{ij} F_{ij} h_{ij}^2$, and so $\bar{m}$ tends to increase.

population of strings just as in the Markov chain, except that it is evolving as if the payoffs were the false ones.

What makes this non-trivial is that there is a relationship between the true and false payoffs. If we change the excess payoff a_i on every state from true excess payoff to false excess payoff, it's as if payoff is transferred from state to state within each amalgamation. The average payoff in each amalgamation is unchanged. In that sense, payoff is conserved. Freeloading strings are stealing payoff that rightly belongs to other strings, much as in evolutionary biology and evolutionary computation. The details of all these matters are given in [3].

All this was outlined by John Holland in his 1975 book, *Adaptation in Natural and Artificial Systems* [1]. The problem back then was that crucial steps in the formalization were missing. To obtain the basic equivalence in theorem 1, we have to think of action values as including pre-values as well as post-values. Without that time-symmetric insight, the equations of learning and evolution remained stubbornly different. The fields of reinforcement learning and evolutionary computation separated and drifted apart, each field quite properly ignoring the other, because each had little to contribute to the other. Until now. Now it is possible to re-establish contact.

In establishing the equivalences that Holland foresaw, it is the temporal difference methods that have been the most recalcitrant. But in paper [3] we have shown that even they yield to the time symmetric approach.

Holland was right.

References

[1] J. H. Holland. *Adaptation in Natural and Artificial Systems.* Univ. of Michigan Press, Ann Arbor, 1975.

[2] Tom Westerdale. Learning Resembles Evolution – the Markov Case. pages 1–12, 2024. unpublished. URL `https://www.dcs.bbk.ac.uk/~tom/briefsymmetric.pdf`.

[3] Tom Westerdale. Learning Resembles Evolution even when using Temporal Diffference. pages 1–16, 2024. unpublished. URL `https://www.dcs.bbk.ac.uk/~tom/briefgeneral.pdf`.