Combining Reinforcement Learning with a Local Control Algorithm

Jette Randløv
University of Copenhagen, Blegdamsvej

Combining Reinforcement Learning with a Local Control Algorithm

Jette Randløv RANDLOV@NBI.DK
CATS, Niels Bohr Institute, University of Copenhagen, Blegdamsvej 17, DK-2100 Copenhagen Ø, Denmark

Andrew G. Barto BARTO@CS.UMASS.EDU
Michael T. Rosenstein MTR@CS.UMASS.EDU
Department of Computer Science, University of Massachusetts, Amherst, MA 01003 USA

Abstract

We explore combining reinforcement learning with a hand-crafted local controller in a manner suggested by the chaotic control algorithm of Vincent, Schmitt and Vincent (1994). A closed-loop controller is designed using conventional means that creates a domain of attraction about a target state. Chaotic behavior is used or induced to bring the system into this region, at which time the local controller is turned on to bring the system to the target state and stabilize it there. We describe experiments in which we use reinforcement learning instead of, and in addition to, chaotic behavior to learn an efficient policy for driving the system into the local controller's domain of attraction. Using a simulated double pendulum, we illustrate how this method allows reinforcement learning to be effective in a problem that cannot be easily solved by reinforcement learning alone, and we show how reinforcement learning can improve upon the chaotic control algorithm when the domain of attraction can only be approximately determined. Similar results are shown using the Hénon map. This is a simple and effective way of extending reinforcement learning to more difficult problems.

1. Introduction

For reinforcement learning (RL) methods to find wider use as on-line methods for improving control performance of real systems it is important to devise methods that take advantage of existing control methodologies to 1) reduce the complexity of the learning problem and 2) to provide for acceptable system behavior during learning. In this paper we explore combining RL with an algorithm proposed by Vincent, Schmitt, and Vincent (1994) that switches between chaotic behavior and a local controller to bring a nonlinear system to a target state and stabilize it there (see also Vincent 1997a, b; Vincent & Grantham 1997).

Vincent's algorithm applies to nonlinear systems that are chaotic or that can be made to produce chaotic behavior by an open-loop control. Called the *chaotic control algorithm*, it works as follows. A closed-loop controller, which we will call the local controller, is designed using standard design methods that is guaranteed to drive the system to the desired target state from any state within some neighborhood of the target and to stabilize the system there. The largest such neighbourhood is called the controllable set of the local controller. If the controllable set intersects the system's chaotic attractor, then starting from any state in the domain of attraction of the chaotic attractor will guarantee that the system will eventually enter the controllable set. The chaotic control algorithm detects when the system enters the controllable set, or a subset thereof, and then turns on the local controller (and at the same time turns off the open-loop controller that may have been used to induce chaotic behavior). Under this closed-loop control, the system then stabilizes at the target state.

The key requirements are that a suitable local controller can be designed such that its controllable set has a non-empty intersection with the chaotic attractor and that entry into the controllable set, or a subset thereof, can be detected. Vincent (1997b) and Vincent and Grantham (1997) start with a system model, linearize it about the target state, and design a linear quadratic regulator (LQR) about the target state using standard LQR methods (e.g., Ogata, 1987). Various methods exist for determining the controllable set, but it is often necessary to settle for an approximation. In some cases only a rough approximation of the controllable set is possible, and it is necessary to turn on the local controller several times before the system is captured to the desired target state, turning it off each time the system exits the putative controllable set (Vincent, Schmitt, & Vincent 1994).

This chaotic control algorithm suggests several ways that RL can be similarly used in conjunction with chaotic behavior and a hand-crafted local controller. Assume that we have been able to design a local controller for a desired target state and that we have a method for determining when the system enters its controllable set (or a reasonably large subset of it). RL can adjust a closed-loop controller for use outside of the controllable set with the objective of reaching the controllable set in minimum time. If the system is chaotic, or is made to exhibit chaotic behavior via an open-loop control, this chaotic behavior can be used as the mode of exploration of the RL system. With this scheme, the prior knowledge embodied in the local controller can make the learning part of the task much easier due to the increased size of the RL system's goal set (now the controllable set of the local controller). At the same time—if the rewards are defined appropriately—RL can improve upon the chaotic control algorithm by learning to avoid states in the putative controllable set of the local controller that are not in its actual controllable set. These are states from which the local controller cannot stabilize the system.

In this paper we describe several experiments motivated by these ideas using a simulated double pendulum motivated by the work of Vincent, Schmitt and Vincent (1994). We then describe experiments with a simpler but naturally chaotic system, the Hénon map, motivated by the work of Vincent (1997b), that allowed us to further explore some of the questions raised by the double pendulum results.

A variety of methods have been proposed for using domain knowledge and/or learned models for improving learning on pendulum swingup tasks related to our double pendulum problem (e.g., Atkeson & Schaal, 1997; Atkeson & Santamaría, 1997; Boone, 1997a,b). Switching control is also a well-known approach to these and other nonlinear control problems (e.g., Spong, 1995; Spong & Praly, 1996), and RL has been used in the control of the Hénon map and other chaotic systems (Gadaleta & Dangelmayr, 1999). However, we are not aware of studies that consider the combination of RL with switching control that we propose here.

2. Double Pendulum

The double pendulum has two links, one attached at the end of the other. At the base of each link is a motor with limited power (Figure 1).

The system's equations of motion are:

$$\begin{aligned}\ddot{\theta}_1 \;=\; & \left[s_1(m_1+m_2\sin^2(\theta_1-\theta_2))\right]^{-1}\\ & \cdot\Big(-s_1 m_2\dot{\theta}_1^2\sin(\theta_1-\theta_2)\cos(\theta_1-\theta_2)\\ & \quad -m_2 s_2\dot{\theta}_2^2\sin(\theta_1-\theta_2)+g m_2\sin\theta_2\cos(\theta_1-\theta_2)\\ & \quad -g(m_1+m_2)\sin\theta_1+\tau_1\Big)\end{aligned}$$

and

$$\begin{aligned}\ddot{\theta}_2 \;=\; & \frac{1}{s_2}\Big(-s_1\ddot{\theta}_1\cos(\theta_1-\theta_2)+s_1\dot{\theta}_1^2\sin(\theta_1-\theta_2)\\ & \quad -g\sin\theta_2+\frac{\tau_2}{m_2}\Big)\end{aligned}$$

where θ_1 and θ_2 are respectively the angles from the first and second links to vertical; $m_1 = 1$ kg and $m_2 = 1$ kg are the respective masses of the first and second links; $s_1 = 1$ meter and $s_2 = 1$ meter are their lengths; and τ_1 and τ_2 are the torques the controller applies at the first and second joints. The maximum torque is 8.5 newtons, so that $\tau_1, \tau_2 \in [-8.5\text{N}, 8.5\text{N}]$.

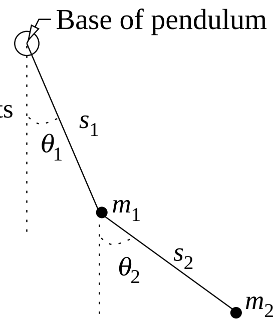

Figure 1. The Double Pendulum. There is a motor at each joint.

The system starts with both links hanging motionless downwards. The objective is to balance both links straight up vertically, i.e., to stabilize about the target state ($\theta_1 = \theta_2 = \pi, \dot{\theta}_1 = \dot{\theta}_2 = 0$). Although the motors generate sufficient torque to produce stability in the neighbourhood of the target state, they do not generate sufficient torque to achieve this target directly from the initial state without following an oscillatory trajectory. It is an underpowered nonlinear system for which no linear controller is globally effective. We simulated this system using an Euler method with a step size of 0.001 seconds.

3. Reinforcement Learning

Reinforcement learning consists of a collection of methods for approximating solutions to stochastic optimal control problems (Sutton & Barto, 1998). These methods adjust a closed-loop control rule, or policy, which is a mapping from system states to control actions. Usually formulated for discrete-time systems with an immediate reward r_{t+1} being delivered to the learning system in response to the execution of control action u_t in state x_t, the most commonly studied objective is to maximize for each time step t the expected discounted return defined to be the discounted sum of rewards over future time steps:

$$\sum_{k=0}^{\infty} \gamma_k r_{t+k+1},$$

where γ, $0 \leq \gamma < 1$ is a discount factor.

In this study, we use the RL algorithm known as Sarsa(λ) (Rummery, 1995; Sutton & Barto, 1998) with replacing eligibility traces (Singh & Sutton, 1996). This algorithm works as follows. Let $Q_t(x,u)$ denote the estimate at time t of the value of the state-action pair (x,u). This is an estimate of the expected return starting from state x, executing control action u, and following an optimal policy thereafter. At each time step, this estimate is updated for all state-action pairs using the update equation

$$Q_{t+1}(x,u) = Q_t(x,u) + \alpha\delta_t e_t(x,u),$$

where δ_t is the temporal difference error at step t:

$$r_{t+1} + \gamma Q_t(x_{t+1}, u_{t+1}) - Q_t(x_t, u_t),$$

and $e_t(s,u)$ is the value of the eligibility trace for state-action pair (x,u) at time t, and α is a positive step-size parameter. The eligibility trace is updated for all state-action pairs (s,u) as follows:

$$e_t(x,u) = \begin{cases} 1 & \text{if } x = x_t \text{ and } u = u_t; \\ \gamma\lambda e_{t-1}(x,u) & \text{otherwise.} \end{cases}$$

where λ, $0 \leq \lambda \leq 1$, is the trace-decay parameter.

The most up-to-date value estimates of the state-action pairs containing the currently observed state x_t are used to determine the control action the learning system takes at time step t. Specifically,

$$u_t = \arg\max_{u'}[Q_t(x,u') + \eta_t],$$

where, for the double pendulum, η_t is a random number drawn uniformly from the interval $[-0.00005, 0.00005]$. The addition of this small noise term produces some exploratory behavior, especially at the beginning of learning when the value estimates are nearly equal. In our experiments, we set $\alpha = 0.1$, $\gamma = 0.98$, and $\lambda = 0.99$.

In applying Sarsa(λ) to the double pendulum, we let the learning algorithm select from a set of nine control actions consisting of the torque pairs (τ_1, τ_2), where $\tau_i = \pm 8.5$ or 0 for $i \in \{1,2\}$. Every 20 time steps of the double pendulum simulation, the learning system selected an action (a pair of torques) and held it constant for 20 time steps. Thus, the time step for learning corresponded to 0.02 sec, whereas the pendulum simulation time step was 0.001 sec. We used a simple rectangular aggregation method to represent the estimated values of the state-action pairs. For each of the nine actions, we created a table with 2,432 entries. Each entry was the value associated with the corresponding action and a rectangular region of state space determined by the following non-overlapping intervals: for each of θ_1 and θ_2: 0, ± 0.6, ± 2.4, $\pm\pi$ radians; and for each of $\dot{\theta}_1$ and $\dot{\theta}_2$: 0, ± 0.25, ± 0.5, $\pm\infty$ radians per second.

One might attempt a simple, pure RL approach to this problem. For example, one could define the reward r_t to be -1 for each time step in which the state of the double pendulum is not within some small region of the target state, and set $r_t = 0$ within this region. This would set up the RL system to attempt to learn to drive the system to this target region in the minimum number of time steps and keep it within that region. This approach has almost no chance of producing an acceptable solution within any reasonable time period. The target state—both links motionless upwards—is a repelling state. For a simple RL system to get close to this state at all would be extremely unlikely. The learning time for such a RL system would be so long that we could say that the task is effectively unlearnable by such a system.

4. A Local Controller

Following the derivation method in Vincent and Grantham (1997) pp. 91–114 and using standard LQR techniques, we obtained a saturating LQR control rule for the double pendulum linearized about the target state. To obtain this control rule, we first found the closed-loop control rule $u = u(x)$ that minimizes, for the linearized system, the following expression:

$$J = \int_0^\infty \left[(x-\bar{x})^T(x-\bar{x}) + u^T u\right] dt, \tag{1}$$

where $\bar{x}$ is the target state. These controls are then restricted to lie in the appropriate region of control space. Specifically, our saturating LQR control rule for the double pendulum is given by

$$\begin{pmatrix}\tau_1\\ \tau_2\end{pmatrix} = \begin{pmatrix} -19.6243 & 8.2425 & -19.6666 & -2.4341 \\ -19.6666 & 3.4245 & 19.6243 & 5.8586 \end{pmatrix} \cdot \begin{pmatrix} \theta_1 - \pi \\ \dot{\theta}_1 \\ \theta_1 - \theta_2 \\ \dot{\theta}_2 \end{pmatrix}$$

provided that $|\tau_i| \leq 8.5$, $i \in \{1,2\}$; otherwise, if $\tau_i > 8.5$ then $\tau_i = 8.5$, and if $\tau_i < -8.5$ then $\tau_i = -8.5$.

This control rule is guaranteed to bring the system to the target state and stabilize it there from any state within its controllable set. Characterizing the actual controllable set is a difficult problem for this system, but there are several ways to approximate it. We followed Vincent, Schmitt and Vincent (1994) and used as an estimate of the controllable set the region in which the angles of both links are less than 0.75 radians away from the target state (i.e., away from π radians). In the chaotic control algorithm the local controller is turned on whenever both link angles are within 0.75 radians of π.

5. The Chaotic Control Algorithm

As mentioned above, the chaotic control algorithm relies on inducing chaotic behavior that causes the system to enter the estimated controllable set of the local controller. When this is detected, the local controller is turned on to stabilize the system about the target state. We induced chaotic behavior in the double pendulum system by driving the it with the following open-loop control signal:

$$\tau_1 = 8.5\cos(1.5\,t) \quad \text{and} \quad \tau_2 = 8.5\cos(0.7\,t)$$

where t is the time in seconds. Of several open-loop controllers with which we experimented, this one is the fastest on average to cause the system to enter the estimated controllable set of the local controller described in Section 4. This open-loop control signal is turned off when the local controller is turned on whenever the system enters the estimated controllable set. However, because there may be states in the estimated controllable set that are not in the actual controllable set, we let the open-loop controller take over again if the system leaves the estimated controllable set.

In our experiments with a number of different open-loop controllers, we found that the number of time steps it takes the chaotic control algorithm to stabilize the system is 1,651 (± 102) on average for a good open-loop controller, and up to 3,000 for a less good one. We used the fastest we found for further experiments. The averages where measured over 50 runs.

6. Combining the Local Controller and RL

First we combined the local controller and the RL controller by letting the RL controller take over the role of the open-loop controller in the chaotic control algorithm: if the RL controller drives the double pendulum into the estimated controllable set, the local controller takes over. If the local controller makes the system leave the controllable set, the RL system regains control.

How should reward signals be supplied to the RL controller to make this scheme work? If we just reward it for getting the double pendulum into the estimated controllable set, then it will be rewarded even if it drives the system to a state in the estimated controllable set that is not in the actual controllable set. In this case, the local controller will not be able to stabilize the system so that the desired outcome will not be achieved. The method we found to work well is to reward the RL controller when the state enters the estimated controllable set, and to punish it slightly more if it has to regain control from the local controller (i.e., if the double pendulum does not stabilize and the RL controller has to take control again). Additionally, when the RL controller is not in control, its learning algorithm is shut off as well. This means that the RL controller experiences the time interval during which the local controller is engaged as if it were a single time step. It may seem a little strange that we punish the RL controller for another controller's failure, but the RL controller is the only adapting component, and therefore the only part that has a chance of avoiding punishment by placing the system into states that are in the actual controllable set.

Table 1 describes the main points in the algorithm for this way of combining the local controller with RL. It describes what happens during each learning *trial*, which begins with the double pendulum in its initial state (both links hanging motionless downwards) and ends when the target state is reached (at which we know the local controller can stabilize the system). A learning *run* begins with initialization of the state-action value function, Q, and the eligibility function, e, to zero, and consists of a large number of learning trials. We adopted the approach commonly used in RL of rewarding the system with a -1 on each time step until a goal state is reached (in this case, until the estimated controllable set is reached) to encourage the system learn to reach the goal in the minimum number of time steps.

Table 1. The main points in the combination of the RL controller and the local controller.

1.	The RL controller generates a control action and updates Q and e.
2.	If the system has *not* entered the estimated controllable set, the RL controller receives a reward of -1. Jump to 1.
3a.	Otherwise the system has entered the estimated controllable set, and the local controller takes over. The RL controller receives a reward of 1.
3b.	The local controller generates the next action.
3c.	If the system exits from estimated controllable set, the RL controller receives a reward of -2. Jump to 1.
3d.	If the system reached the target state, the trial terminates.
3e.	Otherwise jump to 3b.

Figure 2 shows the learning curve for the RL controller combined as described above with the local controller. The number of time steps per trial is the total number including the those during which control is provided by the local controller.

These results show that this way of combining the RL controller with the local controller works very well. From Figure 2 we can see that the RL controller combined in this way with the local controller achieved on average much faster stabilization times—less than 250 time steps—than the 1,651 time steps we achieved for the best chaotic control algorithm. An especially interesting result is that the RL controller learns to avoid states in the estimated controllable set that are not in the actual controllable set of the local controller. Figure 3 shows the number of times the double pendulum entered the estimated controllable set but the local controller was later turned off because it could not actually stabilize the system. From the start of learning,

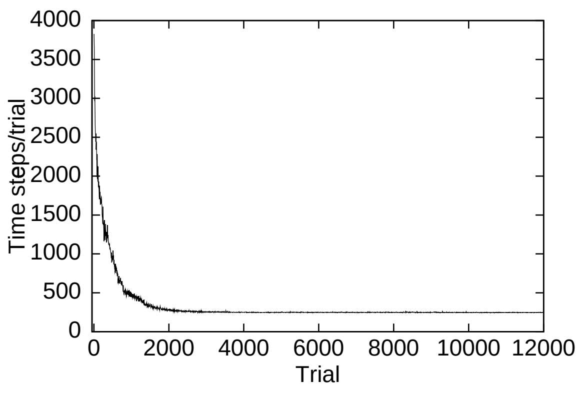

Figure 2. Learning curve for a RL controller combined with the local controller. Average of 100 runs.

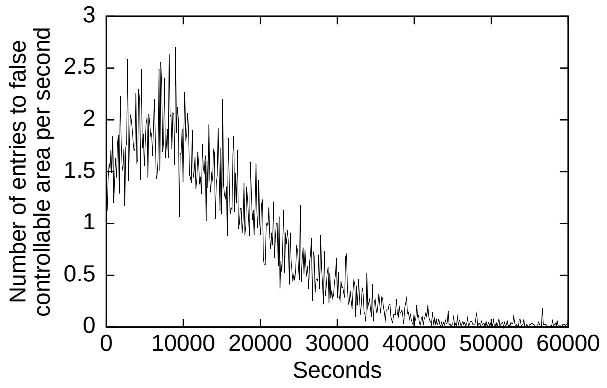

Figure 3. The RL controller learns to avoid states in the estimated controllable set that are not in the actual controllable. One second consists of 50 time steps. Average of 100 runs.

this number increases as the RL controller learns to get the double pendulum into the estimated controllable set, and then it decreases as the RL controller learns which states in this set work and which should be avoided. This improvement is caused by the RL controller being punished for the local controller's failure to stabilize the system.

7. Combining the Chaotic Control Algorithm with RL

We also experimented with combining the RL controller with the chaotic control algorithm, that is, with combining RL with the open-loop controller described in Section 4 that induces chaotic behavior, as well as with the local controller. Here the idea is to let the chaotic behavior provide the exploratory behavior from which the RL controller can learn. One way to do this is to run the RL controller at the same time as the open-loop controller and let both of their control decisions determine the actual control action sent to the double pendulum. We let the control action be a convex linear combination of the control choices of the chaotic and RL controllers, with the coefficient varying during learning to adjust the relative contribution of each controller. At the start of each learning run, control choices were due entirely to the open-loop control rule, while the RL controller controller learned as if these were its own decisions. As each learning run proceeded, we altered the coefficient so that an increasing proportion of the control was due to the RL controller's choices.

We experimented with several methods for decaying the influence of the open-loop controller. Our best results were obtained with a linear and an exponential decay. Specifically, for the linear method

$$\omega = \min\{1, \tfrac{t}{1000}\}$$

and the control action is:

$$\tau_i = \omega \tau_i^{\mathrm{RL}} + (1-\omega)\tau_i^{\text{open-loop}}$$

for $i \in \{1,2\}$. For the exponential decay

$$\omega = \min\left\{1, \exp\left(-\frac{t-500}{200}\right)\right\}$$

and

$$\tau_i = (1-\omega)\tau_i^{\mathrm{RL}} + \omega \tau_i^{\text{open-loop}}$$

for $i \in \{1,2\}$, where t is the trial number in a learning run.

Figure 4 shows our best results averaged over 100 runs. At the start of each run, the control actions were due entirely to the open-loop controller. As the number of trials increases, the RL controller's actions contribute an increasing proportion. Comparing Figure 4 with Figure 2, one sees that both methods for phasing in RL eliminate the very long trials at the beginning of learning, but they add more area under the learning curve. In total, the result is a much larger number of time steps before a good level of performance is reached than achieved using RL alone outside of the estimated controllable set.

8. Improving over the Local Controller in terms of Time-to-Target

The local controller is not a minimum-time controller (it minimizes a measure that combines distance from the target and the motor force needed). Therefore theoretically it should be possible for the RL controller to do better than the local controller in terms of the time taken to reach the target state. In terms of the performance level eventually achievable, it might be better to use a *smaller* estimated controllable set than a larger one, since this would decrease the region in which the non-minimum-time local controller would operate. To study this possibility, we experimented with decreasing the estimated controllable set during learning, and then compared the local controller's original performance with that of this new combination that has more freedom of action near the target state.

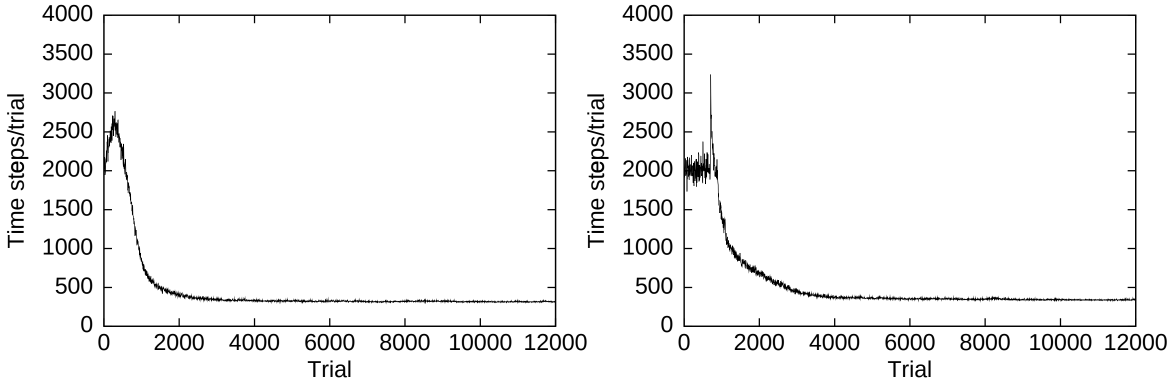

Figure 4. Learning curves for combinations of the chaotic and RL controllers for the double pendulum. For the left graph, the weighting of the chaotic controller decreases linearly, and for the right graph it decreases exponentially. Both graphs are averages of 100 runs.

Specifically, we used the combined RL and local controllers as described in Section 6 for the first 12,000 trials, activating the local controller if both links of the double pendulum got within 0.75 radians from the target state.[1] We decreased this activation angle during trials 12,000 to 24,000 linearly from 0.75 to 0.01 radians, at which setting the RL controller was allowed to learn for an additional 12,000 trials.

After the first 12,000 trials the average number of time steps from the time that the 0.75 radian criterion was met for the last time until the target was reached, was 70.12 (± 0.03). This number is the local controller's original performance. After decreasing the estimated controllable set and learning for 24,000 additional trials, the number of time steps had dropped to 69.88 (± 0.02). These numbers are based on 100 runs and trials 11,900–12,000 and 35,900–36,000. This improvement is not very large, but it shows that the RL controller can learn a control rule that can improve on a carefully hand-crafted system-specific solution.

9. Hénon Map

To further explore some of the observations described above, we followed Vincent(1997b) and conducted additional experiments using the Hénon map (Hénon, 1976), one of the simplest systems to exhibit a chaotic attractor. This system has a two-dimensional state space which allowed us to more easily visualize the results of various control methods. It is given by

$$\begin{aligned} x_1(t+1) &= -1.4x_1^2(t) + x_2(t) + 1 \\ x_2(t+1) &= 0.3x_1(t) \end{aligned}$$

and is known to have an unstable fixed point $\bar{x} = [0.6314, 0.1894]^T$.

[1]We used a finer quantization of the state space for this experiment than that described in Section 3. The quantization thresholds for θ_1 and θ_2 were 0, ± 0.6, ± 2.4, ± 2.8, $\pm \pi$ radians.

We implemented Vincent's (1997b) saturating LQR control rule to stabilize the system about $\bar{x}$. A one-dimensional control signal, $u(t)$, was added to the equation for x_1, and the control rule was derived from a local linearization of the system about $\bar{x}$ in a manner analogous to that described above for the double pendulum. The controls were restricted to a interval $[-u_m, u_m]$. We used $u_m = 0.1$. To investigate the ability of RL to learn to avoid subregions of an estimated controllable set, we purposefully used an overly-large estimate of the controllable set given by a circular region of radius 0.25 centered at $\bar{x}$.

We implemented several control schemes for stabilizing the Hénon map at $\bar{x}$. In all cases, we initiated trials by randomly selecting initial states according to a circular Gaussian distribution with standard deviation 0.1 centered at $[-0.2, 0.15]^T$. Trials ended when the state entered a target region of radius 0.025 about $\bar{x}$ or when it exited the region $-1.5 \leq x_1 \leq 1.5$ and $-0.4 \leq x_2 \leq 0.4$ (Figure 7). We called these latter trials 'failures' (although it is possible that some of these trials, if continued long enough, would have re-entered the region of interest). The control schemes were:

LQR: global use of the saturating LQR controller. Although this controller cannot stabilize the system from arbitrary starting states even with no constraints on the control magnitude (Vincent, 1997b), its performance provides a useful baseline.

Chaos+LQR: saturating LQR control within the estimated controllable set, and no control ($u = 0$) outside of this set (the chaotic control algorithm of Vincent et al., 1994).

RL: global use of RL. We used Sarsa(0) with a reward of -1 on each time step until a trial ended, with an additional reward of -1000 if the trial was a failure. The actions were $-u_m$, 0, or u_m, so that the extreme values coincided with LQR's saturated controls. The state space was represented by a lookup table aggregating states into rectangular regions determined by a 50×50 grid over the region $-1.5 \leq x_1 \leq 1.5$ and $-0.4 \leq x_2 \leq 0.4$. We set $\alpha = 1$, a value determined by experiment to work well for this problem when RL was combined with the local controller. We also set $\gamma = 1$ since the trials were of finite duration, and we did not use random exploration, relying instead on the system's chaotic behavior. Note that global RL does not necessarily stabilize the system at $\bar{x}$.

RL+LQR: saturating LQR within the estimated controllable set and Sarsa(0) outside of it (implemented as for global RL). Additional rewards were determined according to the scheme given in Table 1.

Figure 5 gives performance or learning curves for these controllers. Significant improvement is apparent for the RL controllers, especially for RL+LQR. Hitting the small target region with chaos alone took on average 238 (± 8.83) steps; RL+LQR learned to take on average 6.1 (± 0.06) steps. Not shown is the decrease with learning in the number of failure trials (e.g., from an average of 32.5 (± 1.8) over 10 runs of trials 1–100 of RL+LQR, to an average of 1.0 ($\pm .31$) over 10 runs of trials 4900-5000). Figure 6 shows that, as in the double pendulum experiments, RL+LQR learns to reduce the frequency with which trajectories reach states in the estimated controllable set that are not in the actual controllable set. Figure 7 shows the policy learned by RL+LQR. The shaded squares are those visited during learning, and their grey levels code the action selected by the learned control rule. Within the estimated controllable set only the LQR actions are selected. The black squares outside of the estimated controllable set are those in which the learned control rule's actions happened to coincide with those that the saturating LQR controller would have generated in those states. This allows one to see how the learned control rule differs from the global application of the saturating LQR rule.

10. Discussion and Conclusion

The studies here were inspired by the chaotic control algorithm of Vincent, Schmitt, and Vincent (1994), which suggested that a similar combination of RL with a local controller might increase the range of applicability of RL methods. Exploring this using a simulated double pendulum and the Hénon map, we observed the following results. First, using an RL controller alone until control shifted to the local controller was able, after leaning, to achieve stabilization significantly faster than could be achieved by the best chaotic control algorithm we tried. This is not a surprising result because the chaotic control algorithm does not improve its performance through learning.

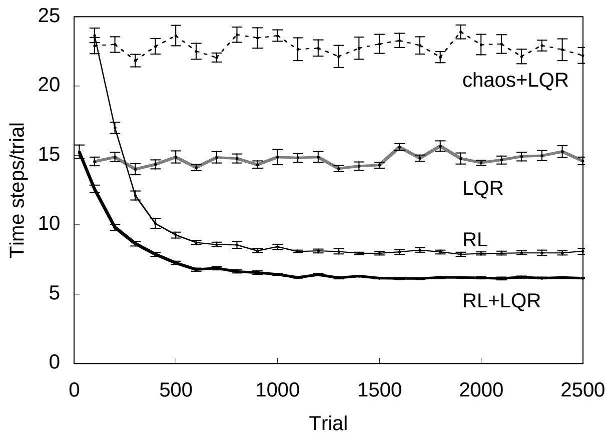

Figure 5. Learning curves for combinations of the chaotic and RL controllers for the Hénon map. Graphs are averages of 10 runs.

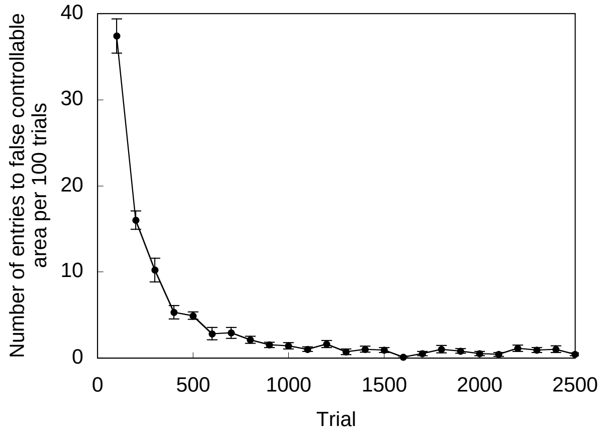

Figure 6. The controller RL+LQR learns to avoid states in the estimated controllable set that are not in the actual controllable set. Average of 10 runs.

A second result that is somewhat more surprising is that letting induced chaotic behavior of the double pendulum guide the exploratory behavior of the RL controller can reduce the length of initial learning trials, but does not necessarily reduce the total time needed to learn a good global control rule. One possible explanation is that the RL controller is learning to compensate for the chaotic component of the control action and has to continue changing its policy as the weight of the chaotic component decreases. Of course, we experimented with just a few methods for integrating RL with chaotic behavior, and many others are possible.

A third result—and perhaps the most interesting—was observed with both the double pendulum and Hénon map. The RL controller learned to compensate for our overestimation of the local controller's controllable set. We observed that the RL controller learned to avoid driving the system to states in the estimated controllable set from which the local controller could not actually stabilize the

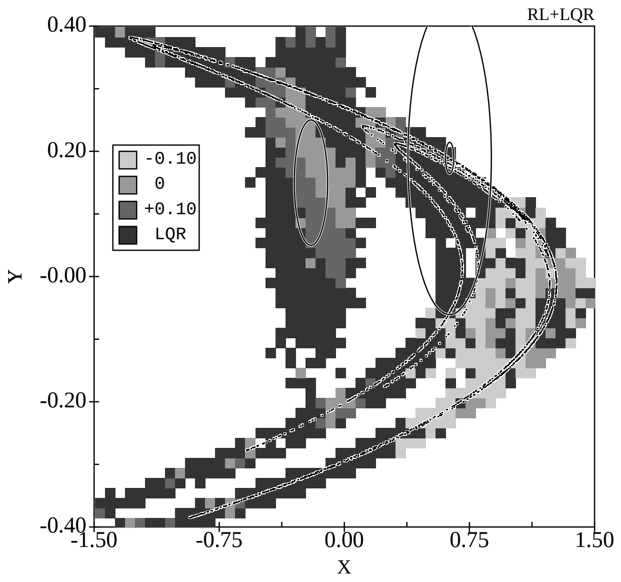

Figure 7. The Hénon state space. The estimated controllable set is shown as the large ellipse centered on $\bar{x} = [0.6314, 0.1894]^T$. The small ellipse is 1 standard deviation of the initial state distribution from $[-0.2, 0.15]^T$. Grey levels code the control rule learned by RL+LQR.

system. This is important because the most difficult aspect of the chaotic control algorithm, and our adaptation of it, is the accurate estimation of the local controller's controllable set. Our results suggest that it may be a good strategy to make a fairly coarse approximation of the controllable set and let learning adjust for its inaccuracies. Further research is needed to determine how learning time changes with increasingly large estimates of the controllable set. Finally, our observations from successively shrinking the estimated controllable set while learning suggest that this might be a good method for compensating for sub-optimal performance of the local controller. But here also further research is needed to determine if substantial improvements can be obtained in this way.

Overall, these results provide additional evidence that conventional control methodology provides a rich avenue for injecting prior knowledge into RL systems and that RL can help improve the utility of conventional control methods when extended to complex nonlinear control problems.

Acknowledgements

We would like to thank Adelle Coster and Preben Alstrøm for comments and interesting discussions, especially about reinforcement learning interacting with constraints. We would also like to thank the anonymous reviewers for their constructive comments and pointers to relevant literature. This project was funded by the Air Force Office of Scientific Research, Bolling AFB (AFOSR F49620-96-1-1054) and the National Science Foundation under Grant No. ECS-9980062. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Atkeson, C. G. and Santamaria, J. C. (1997). A comparison of direct and model-based reinforcement learning. In *International Conference on Robotics and Automation*, pages 3557–3564.

Atkeson, C. G. and Schaal, S. (1997). Robot learning from demonstration. In Fisher, D. H., editor, *Machine Learning: Proceedings of the Fourteenth International Conference*, pages 12–20. Morgan Kaufmann.

Boone, G. (1997a). Efficient reinforcement learning: Model-based acrobot control. In *International Conference on Robotics and Automation*, pages 229–234.

Boone, G. (1997b). Minimum-time control of the acrobot. In *International Conference on Robotics and Automation*, pages 3281–3287.

Gadaleta, S. and Dangelmayr, G. (1999). Optimal chaos control through reinforcement learning. *Chaos*, 9:775–788.

Hénon, M. (1976). A two-dimensional map with a strange attractor. *Communications of Mathematical Physics*, 50:69.

Ogata, K. (1987). *Discrete-Time Control Systems*. Prentice Hall, Englewood Cliffs.

Rummery, G. A. (1995). *Problem Solving with Reinforcement Learning*. PhD thesis, Cambridge University Engineering Department.

Singh, S. P. and Sutton, R. S. (1996). Reinforcement learning with replacing eligibility traces. *Machine Learning*, 22:123–158.

Spong, M. W. (1995). The swing up control problem for the acrobot. *IEEE Control Systems Magazine*, 15:49–55.

Spong, M. W. and Praly, L. (1997). Control of underactuated mechanical systems using switching and saturation. In Morse, A. S., editor, *Control Using Logic Based Switching*, pages 162–172. London: Springer-Verlag. Lecture Notes in Control and Information Sciences 222.

Sutton, R. S. and Barto, A. G. (1998). *Introduction to Reinforcement Learning*. MIT Press/Bradford Books.

Vincent, T. L. (1997a). Control using chaos. *IEEE control Systems*, pages 65–76.

Vincent, T. L. (1997b). Controllable targets near a chaotic attractor. In Judd, K., Mees, A., Teo, K. L., and Vincent, T. L., editors, *Control and Chaos*, pages 260–276. Birkhauser.

Vincent, T. L. and Grantham, W. J. (1997). *Nonlinear and Optimal Control Systems*. John Wiley & Son Inc.

Vincent, T. L., Schmitt, T. J., and Vincent, T. L. (1994). A chaotic controller for the double pendulum. In *Mechanics and Control: Proceedings of the 5th Workshop on Control Mechanics*, pages 257–273.