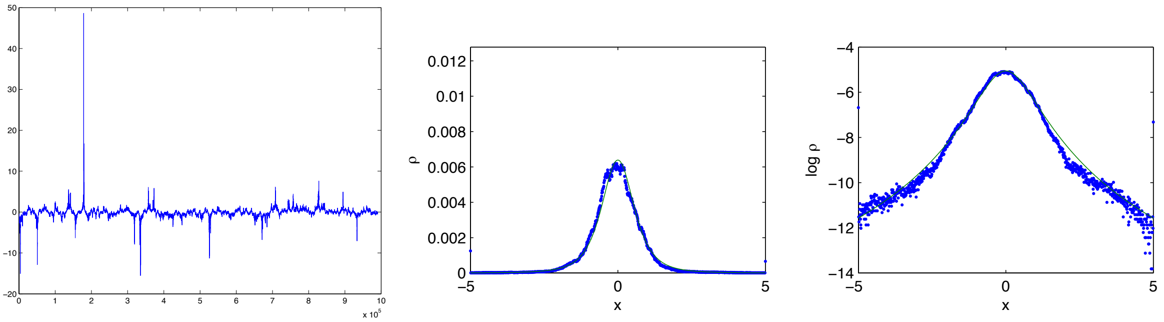

Figure 40: Time series of $dx = d\xi$ with $\langle d\xi^2 \rangle = g(x)dt$, with $g(x) = 1+x^2$. Left time series has large deviations. Right: stationary distribution coincides with $1/g(x)^2$

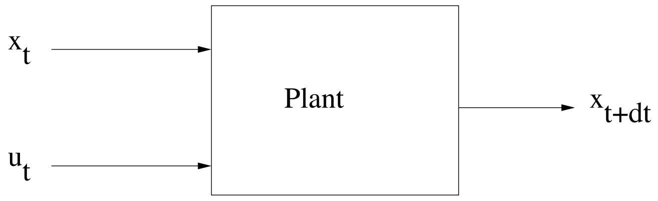

Figure 41: Reinforcement learning scenario. At each time step a control input u_t is given that moves the plant from the current state x_t to a new state x_{t+dt}. The state space is observed. In addition, at each iteration an immediate reward $V(x_t)$ is observed.

In order to verify this example, consider the case $J = 1$, $g(x) = 1 + x^2$ and $R = 1$. The choice of g is to ensure that g becomes nowhere zero which would imply $dx = 0$. Then the dynamics is

$$dx = d\xi' \qquad \langle d\xi'^2 \rangle = \lambda g^2(x)dt$$

which is a random walk with state dependent noise that increases for larger x. We predict the equilibrium solution

$$\tilde{\rho}(x) \propto \frac{1}{g^2(x)}$$

which has heavy tails. Note, that for $g(x) = 1$ we predict a flat distribution, as we expect. It is somewhat surprising that the effect of state dependent noise is a more concentrated stationary distribution. In fig. **??** we show an example of the time series and stationary distribution for $g(x) = 1 + x^2$.

21 Path integral reinforcement learning

When the environment and the intrinsic dynamics are unknown, we can define a learning method similar to RL. We call it Path Integral Reinforcement Learning (PIRL). We consider the following scenario as depicted in Fig. **??**. We must control a plant with unknown internal dynamics and an unknown rewards/costs. At each time step $t = 1, \ldots$ we observe x_t and the immediate reward $V(x_t)$ and can choose a control value u_t and make a dynamics step according to Eq. **??**. Thus, we generate the trace of observed data:

$$x_1, V(x_1), u_1, d\xi_1, x_2, V(x_2), u_2, d\xi_2, x_3, V(x_3), \ldots$$

where the subscripts on $u, d\xi, x$ are in units of dt. u_t are not the optimal controls, but just some trial values. Our goal is to choose u_t such that we learn about the dynamics and the cost, such that we can converge to optimal values of u as fast as possible.

Consider the standard path integral control problem at time t in state x_t

$$dx_i = f_i(x,t)dt + \sum_a g_{ia}(x,t)(u_a(x,t)dt + d\xi_a(t)) \qquad (308)$$

$$C = \left\langle \phi(x_T) + \sum_{s=t}^{T-1} dt \left(\frac{1}{2} u_s^T R u_s + V(x_s, s) \right) \right\rangle^{41} \tag{309}$$

If the u_t are all zero, the optimal control at (x,t) is given by Eq. **??**

$$u_a dt = \langle d\xi_a \rangle (x,t) \tag{310}$$

and the expectation value is wrt to Eq. **??**

$$\begin{aligned} p(x_{t+1:T}|x,t) &= \frac{1}{\psi(x,t)} \prod_{s=t}^{T-1} p(x_{s+1}|x_s) \exp(-\phi(x_T)/\lambda) \\ p(x_{s+1}|x_s) &= \left(\frac{1}{2\pi}\right)^{n/2} \frac{1}{\sqrt{\det \Xi(x_s)}} \\ &\exp\left(-\frac{1}{2\lambda dt} (x_{s+1} - x_s - f(x_s,s)dt)^T \Xi(x_s)^{-1} (x_{s+1} - x_s - f(x_s,s)dt) - V(x_s,s)dt/\lambda \right) \end{aligned}$$

We can compute $\langle d\xi_a \rangle (x,t)$ using the uncontrolled dynamics. We generate a number of trajectories $x^\mu_{t:T}$ using noise $d\xi^\mu_{t:T-1}$, all initialized at $x_t = x$, then

$$\begin{aligned} S^\mu(x,t) &= \sum_{s=t}^{T-1} V(x^\mu_s, s)dt + \phi(x^\mu_T) \\ J(x,t) &= -\lambda \log \frac{1}{N} \sum_\mu \exp(-S^\mu(x,t)/\lambda) \\ \langle d\xi \rangle (x,t) &= \frac{\sum_\mu d\xi^\mu_t \exp(-S^\mu(x,t)/\lambda)}{\sum_\mu \exp(-S^\mu(x,t)/\lambda)} \end{aligned}$$

where N is the number of sample trajectories. This is quite an important result. It means that when $d\xi$ is observed, the optimal cost-to-go and the optimal control can be estimated by sampling from the uncontrolled dynamics using a simulator without actually having to estimate f_i and g_i.

If we dont observe $d\xi_a$, we can use the uncontrolled dynamics $dx_i = f_i(x,t)dt + g_{ia}(x,t)d\xi_a$ to write Eq. **??** as

$$\langle dx_i \rangle = f_i(x,t)dt + g_{ia}(x,t)u_a(x,t)dt \tag{311}$$

By computing local estimates of $\langle dx_i \rangle, f_i(x,t)$ and $g_{ia}(x,t)$ we get an estimate of the optimal control $u_a(x,t)$.

We can improve the computation by using importance sampling. We generate a number of trajectories $x^\mu_{t:T}$ and controls $u^\mu_{t:T-1}$ and noise $d\xi^\mu_{t:T-1}$ initialized at $x^\mu_t = x$. Denote $y_{t:T} = x_{t:T}, u_{t:T-1}, d\xi_{t:T-1}$, then

$$\begin{aligned} S(y_{t:T}, t) &= \sum_{s=t}^{T-1} V(x_s,s)dt + \phi(x_T) + \sum_{s=t}^{T-1} \frac{dt}{2} (\dot{x}_s - f(x_s,s))^T \Xi^{-1}(x_s)(\dot{x}_s - f(x_s,s)) \\ &\quad - \frac{dt}{2} (\dot{x}_s - f(x_s,s) - g(x_s,s)u_{s+1})^T \Xi^{-1}(x_s)(\dot{x}_s - f(x_s,s) - g(x_s,s)u_{s+1}) \\ &= \sum_{s=t}^{T-1} V(x_s,s)dt + \phi(x_T) + d\xi_s^T R u_s + \frac{dt}{2} u_s^T R u_s \\ \langle d\xi \rangle (x,t) &= \frac{\sum_\mu (u^\mu_t dt + d\xi^\mu_t) \exp(-S(y^\mu_{t:T}, t)/\lambda)}{\sum_\mu \exp(-S(y^\mu_{t:T}, t)/\lambda)} \end{aligned} \tag{312}$$

with $\dot{x}_t = (x_{t+dt} - x_t)/dt$ and where we have used the dynamical equations $\dot{x}_s - f(x_s,s) - g(x_s,s)u_s = g(x_s,s)d\xi_s/dt$ to eliminate f, g.

In the last line, we have used the reasoning as in section **??**. We extend the $m \times m$ matrix R to $n \times n$ with diagnonal contributions $R_{ij} = R_\infty \delta_{ij}, i, j = m+1, \ldots, n$. We extend the $n \times m$ matrix g to $n \times n$ with vectors $g_{ij}, j = m+1, \ldots, n$ such that g is of rank n. Then it is easy to show that $(gu_0)^T \Xi^{-1} g u_0 =$

$(u_0)_a R_{ab} (u_0)_b + \sum_{i=m+1}^n (u_0)_i^2 R_\infty$. in the limit of $R_\infty \to \infty$, $u_0 \to 0$ so that we can ignore the second term. Note, that we can use importance sampling also without estimating f, g.

In order to use Eq. **??**, one should in principle compute trajectories forward in time from any state that is visited. This is time-consuming. Instead, one can at each time t compute controls at x from the previously visited states x_t^μ.

The simplest approximation is to ignore the difference between x and x_t^μ which is equivalent to assuming that in the area populated by x, x_t^μ the control is state independent. A further refinement can be made by fitting a linear model, either including all x_t^μ or nearby points only. In either case, this approach yields controls $u_t(x), t = 1, \ldots, T$. These controls can be used in a subsequent importance sampling.

When u is independent of μ, the update equation for u becomes

$$u_t^{k+1} dt = u_t^k dt + \eta_k \frac{\sum_\mu d\xi_t^\mu \exp(-S(y_{t:T}^\mu, t)/\lambda)}{\sum_\mu \exp(-S(y_{t:T}^\mu, t)/\lambda)} \tag{313}$$

where k is the iteration index and we have introduced a 'learning rate' η_k. $\langle d\xi \rangle_t^k$ is highly stochastic and the equation converges to an asymptotic solution when we take η_k small, or anneal it to zero.

The algorithm becomes as follows:

- Initialize $u_{1:T-1} = 0$ and choose $x_0^\mu, \mu = 1, \ldots, n_{\rm traj}$ (identical or different from a distribution).
- For $k = 1, \ldots$
  - Run $n_{\rm traj}$ trajectories using the controlled dynamics $u_{1:T}$ and noise ν resulting in $y_{0:T}^\mu, \mu = 1, \ldots, n_{\rm traj}$.
  - Compute the expected trajectory

    $$x_{0:T}^* = \sum_{x_{1:T}} x_{0:T} p(x_{1:T}|x_0)$$

    or the most likely trajectory

    $$x_{0:T}^* = \text{argmax } p(x_{1:T}|x_0)$$

  - Estimate $u(x_t^*, t)$ for each t using Eq. **??**.

21.1 Learning a deterministic plant

Suppose now, that the plant we wish to control is deterministic, ie. noise is zero.

$$dx_i = f_i(x,t)dt + \sum_a g_{ia}(x,t) u_a dt \tag{314}$$

$$C = \int_0^T \frac{1}{2} u^T R u + V(x,t) \tag{315}$$

the problem is to compute the optimal control law $u_a(x)$.

Suppose that we choose random controls from a Gaussian distribution: $u_a dt = d\xi_a$ with $d\xi \sim \mathcal{N}(0, \nu dt)$ with $\nu = \lambda R^{-1}$ and λ some value and with these random controls we sample trajectories using the above dynamics. Eq. **??** becomes

$$dx_i = f_i(x,t)dt + \sum_a g_{ia}(x,t) d\xi_a \tag{316}$$

We can view Eq. **??** as the uncontrolled dynamics of the stochastic control

$$dx_i = f_i(x,t)dt + \sum_a g_{ia}(x,t)(u_a dt + d\xi_a) \tag{317}$$

$$C = \left\langle \int_0^T \frac{1}{2} u^T R u + V(x,t) \right\rangle \tag{318}$$

which is equivalent to the original control problem Eqs. **??**-**??** when $\lambda \to 0$. This is the problem that we considered above and we can obtain a solution for any x by considering sampled trajectories near x either using the uncontrolled dynamics or with importance sampling. Note, that in this setting $d\xi$ is observed, so that we can compute the optimal control without ever estimating f and g.

Note, the role of λ. If λ is large, Eq. **??** will explore very well. The resulting solution, however, is optimal for a very noisy problem, and may be quite suboptimal for the deterministic system. On the other hand, taking λ small, exploration will be quite poor. A good value of λ balances good exploration and good approximation.

Since the problem is to find a deterministic control for a given initial state, we further simplify the algorithm by computing a state-independent control at each time. The reason is that when the noise approaches zero, the volume of states at each time shrinks to a single point. Thus, the optimal control then only depends on time. At intermediate noise levels this control will be guiding the search but sub-optimal due to its state independence. When noise approaches zero this control approaches the optimal value.

We test the time dependent, state independent version for the single quadratic well and for the acrobot problem.

21.1.1 Single well

We consider the one-dimensional time dependent control problem

$$\begin{aligned} dx &= udt + d\xi \\ C &= \frac{a}{2}x_T^2 + \int_0^T dt \frac{R}{2}u_t^2 \end{aligned}$$

The optimal control solution is

$$u_t = \frac{-ax}{R + a(T-t)}$$

We simulate $ntrials = 1000$ trajectories starting at $x_0 = 2 \pm 1$ using the uncontrolled dynamics. The control is computed at each time step t by selecting $n_neighbor = 200$ nearest states (left subfigure). We also implemented a version where states within a ball of fixed size $\rho = 0.1$ are chosen (right subfigure). A local linear model is computed (but the correction do to the local linear model is negligible compared to the average term). $\nu = \frac{x_0^2}{2T}$ is chosen such that the probability at the target is maximal (which is unfair, since the target location is not known, but alas).

We test the algorithm in fig. **??**. Note, the good quality of the control solution (top figure subplot(3,2,5) and bottom right figure subplot(5,2,3)) and state trajectory (bottom right figure subplot(5,2,1)).

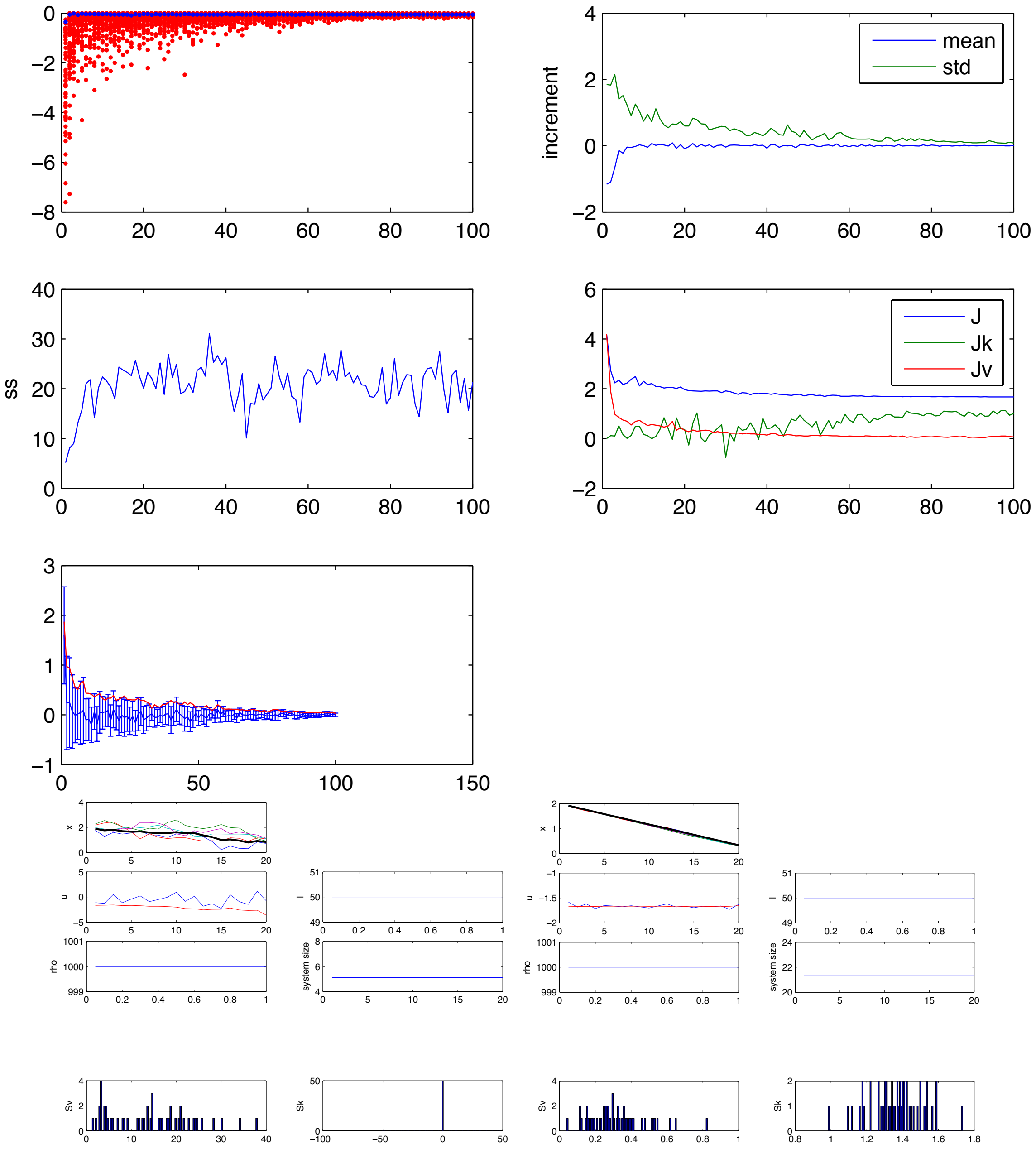

Figure 42: The quadratic control problem with end cost. At each iteration, $n_{\rm traj} = 50$ stochastic trajectories were generated from the same initial starting point $x_0 = 2$. The new control was computed from a deterministic trajectory (starting from x_0), using Eq. **??** with $\eta_k = 0.5$ constant. A total of $n_{\rm ref} = 100$ iterations were used. Initial noise was $\nu = 1$ which was lowered at each iteration with a factor 0.9. All trajectories were included in the control computation. Horizon time is $T = 1$, control cost $R = 1$, endcost multiplier $a = 5$. TOP FIGURE: Top left: $-V(x_T)/a$ (final height) for all $n_{\rm traj}$ solutions for each iteration $k = 1, \ldots, n_{\rm ref}$ (red) and $-V(x_T)/a$ for deterministic solution (blue). Top right: mean and std of $\langle d\xi \rangle_k^t$ over time, for each k. Second row left: sample size versus iteration. Second row right: expected total cost to go (J) and end cost (Jphi) versus iteration. Bottom row: mean and variance in the error in u_t^k versus k (blue) and total quadratic error (red). BOTTOM FIGURE: Control solution after the first iteration (left) and after the last iteration (right). Each figure consists of 8 panels, which give 1) ten best trajectories $x_1(t)$ (color) and deterministic trajectory (black) 2) same for $x_2(t)$ 3) $u(t)$ (blue) and optimal control (red) 4) number of trajectories considered in Eq. **??** 5) size of neighborhood for inclusion of nearby states (all in this case) 6) effective sample size versus time 7) height of the ten best acrobot solution versus time 8) $x_1(t)$ vs. $x_2(t)$ for 10 best solutions 9) histogram of state dependent part of action S for $n_{\rm traj}$ trajectories 10) histogram of control part of action S for $n_{\rm traj}$ trajectories

21.1.2 Acrobot

We use the definition of the acrobot as in [**?**].

$$
\begin{aligned}
&& d_{11}(q)\ddot{q}_1 + d_{12}(q)\ddot{q}_2 + h_1(q,\dot{q}) + \phi_1(q) = 0 \\
&& d_{21}(q)\ddot{q}_1 + d_{22}\ddot{q}_2 + h_2(q,\dot{q}) + \phi_2(q) = u \\
d_{11} &=& m_1 l_{c1}^2 + m_2(l_1^2 + l_{c2}^2) + I_1 + I_2 + 2m_2 l_1 l_{c2}\cos q_2 = c_1 + 2\cos q_2 \\
d_{22} &=& m_2 l_{c2}^2 + I_2 = 1.33 \\
d_{12} &=& d_{21} = m_2 l_{c2}^2 + I_2 + m_2 l_1 l_{c2}\cos q_2 = d_{22} + \cos q_2 \\
c_1 &=& 2.663 \\
h_1 &=& -m_2 l_1 l_{c2}\sin(q_2)\left(\dot{q}_2^2 + 2\dot{q}_1\dot{q}_2\right) \\
h_2 &=& m_2 l_1 l_{c2}\sin(q_2)\dot{q}_1^2 \\
\phi_1 &=& (m_1 l_{c1} + m_2 l_1)G\cos(q_1) + m_2 l_{c2} G\cos(q_1+q_2) \\
\phi_2 &=& m_2 l_{c2} G\cos(q_1+q_2)
\end{aligned}
$$

where we have used the parameter values: $m_1 = m_2 = 1, l_1 = 1, l_2 = 2, l_{c1} = 0.5, l_{c2} = 1, I_1 = 0.083, I_2 = 0.33, G = 9.8$.

The determinant of the matrix d is given by

$$D = d_{11}d_{22} - d_{12}^2 = c_1 d_{22} - d_{22}^2 - \cos^2 q_2 = 1.77289 - \cos^2 q_2$$

which is always positive. Its inverse is

$$d^{-1} \quad = \quad \frac{1}{D}\begin{pmatrix} d_{22} & -d_{12} \\ -d_{12} & d_{11} \end{pmatrix}$$

Thus, the equations of motion become

$$\begin{pmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{pmatrix} \quad = \quad -d^{-1}\begin{pmatrix} h_1 + \phi_1 \\ h_2 + \phi_2 - u \end{pmatrix} = \frac{1}{D}\begin{pmatrix} -d_{22}(h_1+\phi_1) + d_{12}(h_2+\phi_2) \\ d_{12}(h_1+\phi_1) - d_{11}(h_2+\phi_2) \end{pmatrix} + \frac{u}{D}\begin{pmatrix} -d_{12} \\ d_{11} \end{pmatrix}$$

Note, that the term multipying u depends on q_2.

We can write these equations in standard form and introduce noise as

$$dx_i = f_i(x)dt + g_i(x)(udt + d\xi)$$

with $x_1 = q_1, x_2 = q_2, x_3 = \dot{q}_1, x_4 = \dot{q}_2$ and

$$
\begin{array}{ll}
f_1(x) = x_3 & g_1(x) = 0 \\
f_2(x) = x_4 & g_2(x) = 0 \\
f_3(x) = \frac{-d_{22}(h_1+\phi_1)+d_{12}(h_2+\phi_2)}{D} & g_3(x) = -\frac{d_{12}}{D} \\
f_4(x) = \frac{d_{12}(h_1+\phi_1)-d_{11}(h_2+\phi_2)}{D} & g_4(x) = \frac{d_{11}}{D}
\end{array}
$$

We use as cost

$$V(x) \quad = \quad -a(l_1 \sin x_1 + l_2 \sin x_2)$$

which is minimal when both joints are up vertically. We can

We test the algorithm in fig. **??** in the case of endcost: $C = \left\langle V(x_T) + \int dt \frac{1}{2} R u_t^2 \right\rangle$. The variance in $\langle d\xi \rangle$ is an indicator of the non-smoothness of the control and decreases with iteration (compare initial and final solution of u_t).

The solution in fig. **??** is a not optimal. Running the algorithm with different random seeds we show in fig. **??** another solution which has significantly lower total cost ($J = -77$ in fig. **??** vs $J = -112$ in fig. **??**). Whereas the first solution goes up in one swing, the second solution makes use of the inertia by swinging back and forth once before reaching the goal state.

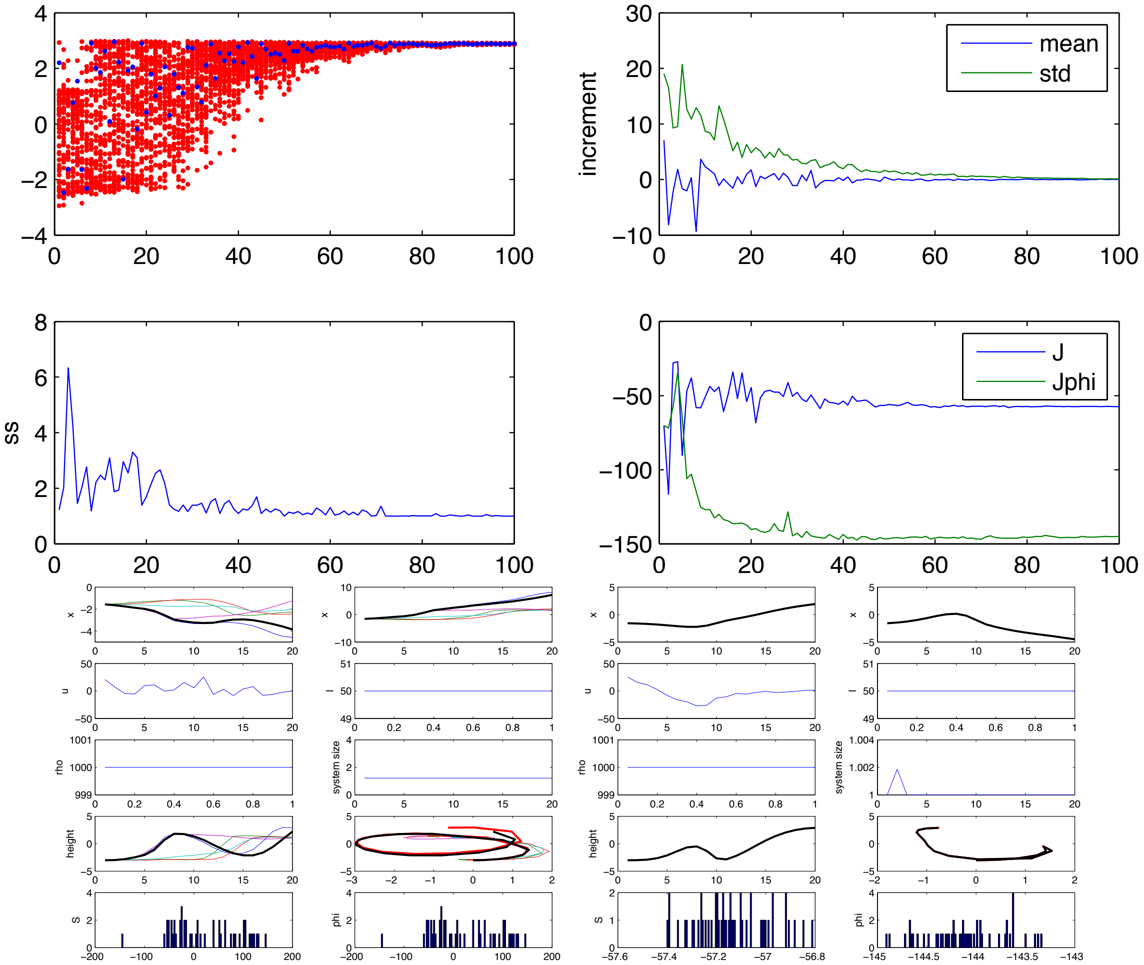

Figure 43: The Acrobot problem with end cost. At each iteration, $n_{\mathrm{traj}} = 50$ stochastic trajectories were generated from the same initial starting point $x_0 = x[-\pi/2, -\pi/2, 0, 0]$. The new control was computed from a deterministic trajectory (starting from x_0), using Eq. **??** with $\eta_k = 0.5$ constant. A total of $n_{\mathrm{ref}} = 100$ iterations were used. Initial noise was $\nu = 20$ which was lowered at each iteration with a factor 0.95. All trajectories were included in the control computation. Horizon time is $T = 1$, control cost $R = 1$, endcost multiplier $a = 50$. TOP FIGURE: Top left: $-V(x_T)/a$ (final height) for all n_{traj} solutions for each iteration $k = 1, \ldots, n_{\mathrm{ref}}$ (red) and $-V(x_T)/a$ for deterministic solution (blue). Top right: mean and std of $\langle d\xi \rangle_k^t$ over time, for each k. Bottom left: sample size versus iteration. Bottom right: expected cost to go versus iteration. BOTTOM FIGURE: Control solution after the first iteration (left) and after the last iteration (right). Each figure consists of 8 panels, which give 1) ten best trajectories $x_1(t)$ (color) and deterministic trajectory (black) 2) same for $x_2(t)$ 3) $u(t)$ 4) number of trajectories considered in Eq. **??** 5) size of neighborhood for inclusion of nearby states (all in this case) 6) effective sample size versus time 7) height of the ten best acrobot solution versus time 8) $x_1(t)$ vs. $x_2(t)$ for 10 best solutions 9) histogram of state dependent part of action S for n_{traj} trajectories 10) histogram of control part of action S for n_{traj} trajectories

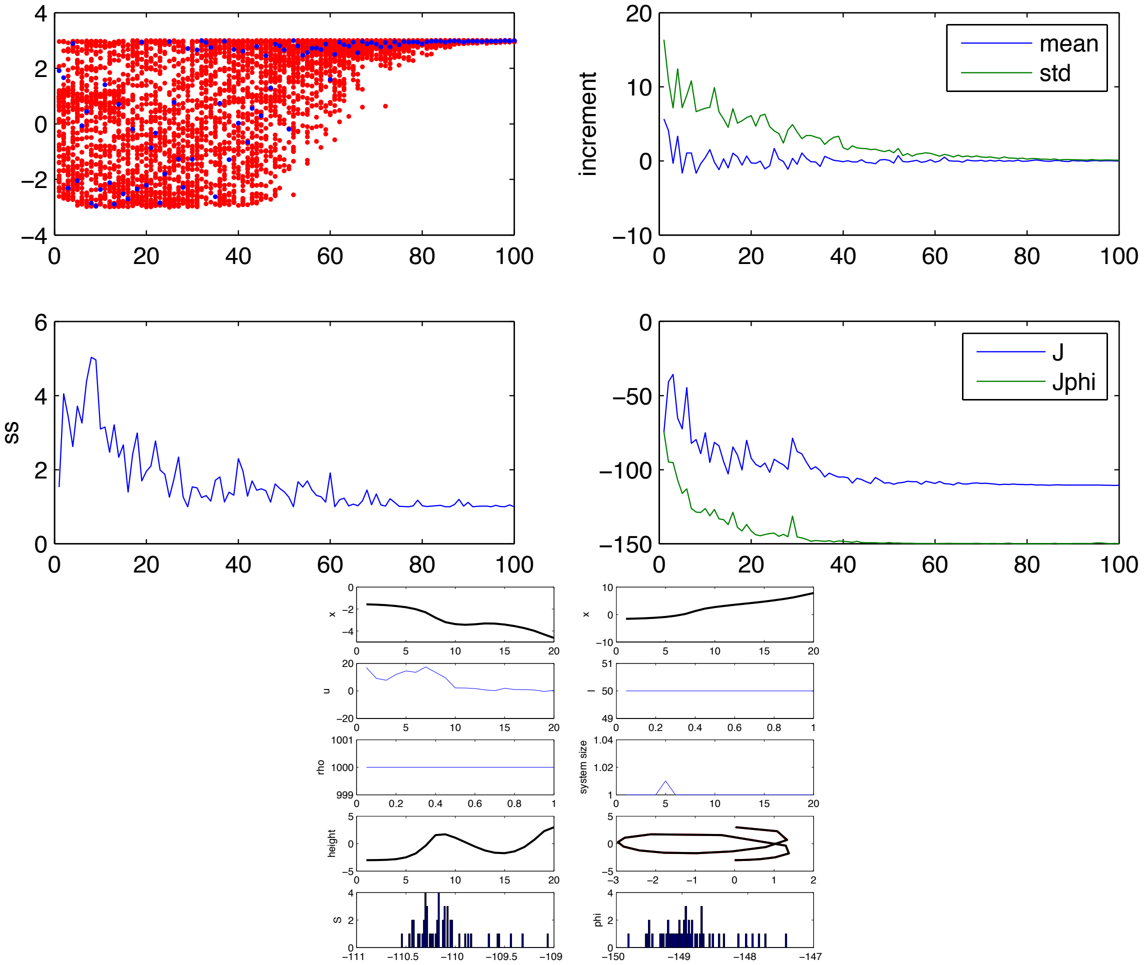

Figure 44: The Acrobot problem with end cost. Different random seed, different solution that is better than the solution of fig. ??. All settings identical.

21.2 Stationary case

We assume that the dynamics is given by Eq. ?? with $f(x)$ and $g(x)$ and $V(x)$ independent of time, $\phi = 0$, $d\xi$ Gaussian noise with zero mean and unknown variance νdt, and u the control to be optimized. R, ν and T are given.

We start from a random initial state and generate a trajectory x_t using controls u_t (initially zero) and noise $d\xi_t$. We denote $y^{t_1:t_2} = (x^{t_1:t_2}, u^{t_1:t_2}, d\xi^{t_1:t_2})$. For each x^t visited, we estimate the optimal control using all nearby past states that have completed their update:

$$\begin{aligned} \mathcal{N}_x &= \{s \in 1 : t-T-1| \, \|x_s - x\| < \rho\} \\ \bar{u}_a^t dt &= \frac{\sum_{s\in\mathcal{N}_x}(u_a^s dt + d\xi_a^s)w^s}{\sum_{s\in\mathcal{N}_x} w^s} \end{aligned}$$

This is a locally constant estimate that ignores the state dependence of the control within $\mathcal{N}_x$.

we can compute a single trace estimate of ψ at that point as

$$\begin{aligned} w^t &= w(x^t) = \exp\left(-S(y^{t+1:t+T})/\lambda\right) && (319) \\ S(y^{t+1:t+T}) &= \sum_{s=t+1}^{t+T} V(x^s)dt + (d\xi^s)^T R u^s + \frac{dt}{2}(u^s)^T R u^s && (320) \end{aligned}$$

where superscripts denote time. This computation involves future states, but can be done on-line by executing at time t:

$$\begin{aligned} w^t &= 1 \\ w^{t-T:t-1} &= w_{t-T:t-1}\exp(-S(y_t)/\lambda) \end{aligned}$$

We get an improved locally linear estimate by making a linear model $u_a^t(x)dt = \sum_i v_{ai}x_i + \theta_a$. We minimize the weighted quadratic criterion

$$\begin{aligned} E &= \frac{1}{2}\sum_{s\in\mathcal{N}_x} w^s \sum_a \left(u_a^s dt + d\xi_a^s - u_a^t(x)dt\right)^2 \\ v_{ai} &= \sum_j \eta_{aj}\chi_{ji}^{-1} \\ \theta_a &= \bar{d\xi}_a - \sum_i v_{ai}\bar{x}_i \\ u_a^t(x)dt &= \bar{u}_a^t dt + \sum_i v_{ai}(x_i - \bar{x}_i) \end{aligned}$$

where we have defined the local statistics

$$\begin{aligned} \bar{x}_i &= \frac{\sum_{s\in\mathcal{N}_x} w^s x_i^s}{\sum_{s\in\mathcal{N}_x} w^s} \\ \chi_{ij} &= \frac{\sum_{s\in\mathcal{N}_x} w^s (x_i^s - \bar{x}_i)(x_j^s - \bar{x}_j)}{\sum_{s\in\mathcal{N}_x} w^s} \\ \eta_{ai} &= \frac{\sum_{s\in\mathcal{N}_x} w^s (d\xi_a^s - \bar{d\xi}_a)(x_i^s - \bar{x}_i)}{\sum_{s\in\mathcal{N}_x} w^s} \end{aligned}$$

Instead of a neighborhood of fixed size ρ, we can define a neighborhood by considering the k states closest to x. In either case, let k be the number of states in $\mathcal{N}_x$.

- If $k = 0$ we can increase ρ upto a preset maximum $\rho_{\max}$. If k is still zero we must generate more sample trajectories
- If $k > 0$, we can compute the above locally constant estimate.
- If $k \geq n$, with n the dimension of x, we can compute a locally linear model using the inverse covariance matrix χ. This is only useful when $\|x - \bar{x}\| < \|\delta x\|$, with x the current point, $\bar{x}$ the (weighted) average position of the k points and δx the variance in their position. We estimate $\|\delta x\|$ by the square root of the smallest eigenvalue of χ.