A Decoupled Data Based Approach to Stochastic Optimal Control Problems

Dan Yu
College of Astronautics
Nanjing University of Aeronautics and Astronautics
Nanjing, China, 210016
yudan198811@hotmail.com

Mohammandhussen Rafieisakhaei
Department of Electrical and Computer Engineering
Texas A&M University
College Station, TX, 77840
mrafieis@tamu.edu

Suman Chakravorty
Department of Aerospace Engineering
Texas A&M University
College Station, TX, 77840
schakrav@tamu.edu

Abstract

This paper studies the stochastic optimal control problem for systems with unknown dynamics. A novel decoupled data based control (D2C) approach is proposed, which solves the problem in a decoupled “open loop-closed loop” fashion that is shown to be near-optimal. First, an open-loop deterministic trajectory optimization problem is solved using a black-box simulation model of the dynamical system using a standard nonlinear programming (NLP) solver. Then a Linear Quadratic Regulator (LQR) controller is designed for the nominal trajectory-dependent linearized system which is learned using input-output experimental data. Computational examples are used to illustrate the performance of the proposed approach with three benchmark problems.

1 Introduction

Stochastic optimal control problems, also known as Markov decision problems (MDPs), have found numerous applications in the Sciences and Engineering. In general, the goal is to control a stochastic system subject to transition uncertainty in the state dynamics so as to minimize the expected running cost of the system. In this work, we propose a novel data based approach to the solution of MDPs residing in continuous state and control spaces. Our approach proposes a rigorous decoupling of the open loop (planning) problem from the closed loop (feedback control) problem in the sense that the decoupled design is near optimal to the third order. Furthermore, this decoupling allows us to propose a highly data efficient approach to solving MDPs in a completely data based fashion (see Table 1 and 2). The approach proceeds in two steps:

• First, we optimize the nominal open loop trajectory of the system using a blackbox simulation model and an NLP solver.
• Second, we identify the linear system governing perturbations from the nominal trajectory using random input-output perturbation data, and design an LQR controller for the linearized system.

2 Related Work

It is well known that the global optimal solution for MDPs can be found by solving the Hamilton-Jacobi-Bellman (HJB) equation \[\theta(x, a) = Q(x, a)\phi(x, a),\] where $\theta$ is the unknown parameter vector, and $\phi$ is a pre-defined basis function, $\phi^\top$ denotes the transpose of $\phi$. Multi-layer (deep) neural networks may also be used as non-linearly parameterized approximators instead of the linear architecture above. The ultimate goal of these techniques is the estimation/learning of the parameters $\theta$ from learning trials/repeated simulations of the underlying system. However, the size of the parameter $\theta$ grows exponentially in the size of the state space of the problem without a compact parametrization of the cost-to-go or Q function in terms of the a priori chosen basis functions for the approximation, and hence, these techniques are typically subject to the curse of dimensionality. Albeit a compact parametrization may exist, a priori, it is usually never known. Recent work on “Deep” RL has shown promise to scale to continuous action and state space robotic learning problems [5][13][18], nonetheless, the amount of training required still seems prohibitive. In the past several years, techniques based on the differential dynamic programming/ILQG approach such as the RL techniques [2][12][13] have shown the potential for RL algorithms to scale to continuous high dimensional robotic task planning and learning problems. For continuous state and control space problems, the method of choice is to wrap an LQR feedback policy around a nominal trajectory and then perform a recursive optimization of the feedback law, along with the underlying trajectory, via repeated simulations/iterations. However, the parametrization can still be very large and can lead to the so-called “policy chatter” phenomenon [2].

Fundamentally, rather than solve the derived “Dynamic Programming” problem as in the majority of the approaches above that requires the optimization of the feedback law, our approach is to directly solve the original stochastic optimization problem in a “decoupled open loop -closed loop” fashion wherein: 1) we solve an open loop deterministic optimization problem to obtain an optimal nominal trajectory in a model free fashion, and then 2) we design a closed loop controller for the resulting linearized time-varying system around the optimal nominal trajectory, again in a model free fashion. Nonetheless, the above “divide and conquer” strategy can be shown to be near optimal to the third order. The primary contributions of the proposed approach are as follows: 1) it shows a
near optimal parametrization of the feedback policy in terms of an open loop control sequence, and
a linear feedback control law, 2) it shows rigorously that the open loop and closed loop learning can
be decoupled, which 3) results in the D2C algorithm that is highly data efficient when compared to
state of the art RL and model-based techniques (Table 1 and 2).

The rest of the paper is organized as follows. In Section 3, the basic problem formulation is out-
lined. In Section 4, a decoupling result which solves the MDP in a “decoupled open loop-closed loop
” fashion is briefly summarized. In Section 5, we propose a decoupled data based control algorithm,
with discussions of implementation problems. In Section 6 we test the proposed approach using
three typical benchmarking examples with comparisons to the model based iLQG approach and
state of the art RL techniques.

3 Problem Setup

Consider the following discrete time nonlinear dynamical system:

\[ x_{k+1} = f(x_k) + B_k(u_k + \epsilon w_k), \]

(1)

where \( x_k \in \mathbb{R}^n \), \( u_k \in \mathbb{R}^m \) are the state and control vector at time \( k \), respectively.
The process noise \( w_k \) is assumed to be zero-mean, uncorrelated Gaussian white noise, with covariance
\( W \), and \( \epsilon \) is a noise scaling parameter.

We consider the following stochastic optimal control problem.

**Stochastic Control Problem:** For the system with unknown nonlinear dynamics \( f(\cdot) \), the optimal
control problem is to find the control policies \( \pi = \{\pi_0, \pi_1, \ldots, \pi_{N-1}\} \) in a finite time horizon
\([0, N]\), where \( \pi_k \) is the control policy at time \( k \), i.e., \( u_k = \pi_k(x_k) \), such that for a given initial state
\( x_0 \), the cost function \( J_\pi = E(\sum_{k=0}^{N-1} c(x_k, u_k) + \phi(x_N)) \) is minimized, where \( \{c(\cdot, \cdot)\}_{k=0}^{N-1} \)
denotes the incremental cost function, and \( \phi(\cdot) \) denotes the terminal cost. The expectation is taken over all
randomness.

4 A Decoupling Result

In the following, we summarize the key “Decoupling” result at the basis of our data based feedback
control design technique. The proofs are shown in Appendix A and Appendix B.

Let \( x'_{k+1} = f(x'_{k}) + B_k(u_k + \epsilon w_k) \) denote the state evolution of the system. Let \( \bar{x}_{k+1} = f(\bar{x}_k) +
B_k \bar{u}_k \) denote the evolution of a nominal state trajectory \( \bar{x}_k \), where \( \bar{u}_k \) is the nominal control action.
Let the incremental cost function have the form \( c(x, u) = l(x) + \frac{1}{2}u^T R u \). Now, let us define a
perturbation of the true trajectory from the nominal, \( \delta x'_k = x'_k - \bar{x}_k \). The evolution of the perturbed
state is given by the equation: \( \delta x'_{k+1} = A_k \delta x'_k + B_k \delta u_k + \epsilon w_k + r_k(\delta x'_k) \), where \( A_k = \frac{\partial f}{\partial x}|_{\bar{x}_k} \),
\( \delta u_k \) is a perturbation in the nominal control effort and \( r_k(\cdot) \) is a residual term containing the second
and higher order dynamics. Let \( J_k(x'_k) \) denote the optimal cost-to-go at time \( k \) from state \( x'_k \). The
optimal cost-to-go function may be expanded in terms of the perturbed state as \( J_k(x'_k) = \bar{J}_k + G_k \delta x'_k +
\delta x'_k P_k \delta x'_k + q_k(\delta x'_k) \), where \( q_k(\cdot) \) denotes the third and third order terms in the cost-
to-go function. Also, we can expand the incremental cost function \( l(x_k) = \bar{l}_k + L_k \delta x'_k + \delta x'_k L_{kk} \delta x'_k +
s_k(\delta x'_k) \).

**Assumption 1** We assume that the functions \( f(\cdot), J_k(\cdot) \) and \( l(\cdot) \) are sufficiently smooth such that
given any nominal trajectory \( (\bar{x}_k, \bar{u}_k) \), they permit expansions in terms of the perturbed state \( \delta x'_k \)
as above. Moreover, we assume that the residual functions \( r_k(\cdot), q_k(\cdot) \) and \( s_k(\cdot) \) in the above expansions are
all uniformly Lipschitz continuous in a neighborhood \( N_k \) of the nominal trajectory \( \bar{x}_k \), for all \( k \).

**Theorem 1 Decoupling.** Let \( \bar{x}_k, \bar{u}_k \) be an optimal nominal trajectory. Under Assumption 1, the
gain \( G_k \) and covariance \( P_k \) evolution equations of the cost-to-go function \( J_k(\cdot) \) are as follows:

\[ G_k = L_k + G_{k+1} A_k, \]

(2)

\[ P_k = L_{kk} + \frac{1}{2} K_k R_k K_k' + A_k' P_{k+1} A_k - K_k' S_k K_k + K_k' B_k' P_{k+1} B_k K_k + G_{k+1} \tilde{R}_{k,xx}, \]

(3)
Further, let \( \hat{J}_k(x_k^i) = \tilde{J}_k + G_k \delta x_k^i + \delta x_k^i P_k \delta x_k^i \). Then, \( |J_k(x_k^i) - \hat{J}_k(x_k^i)| \) is an \( O(||\delta x_k^i||^3) \) function.

**Theorem 2 Large Deviations bound.** Let Assumption \( \mathcal{A} \) holds. Then, there exist finite constants \( \alpha, \beta, \tilde{K} \), independent of \( \epsilon \), such that

\[
\text{Prob}(\max_{0 \leq k \leq N} |\hat{J}_k(x_k^i) - J_k(x_k^i)| \geq \tilde{K}\gamma^3 \epsilon^3) \leq \frac{\alpha}{\gamma} e^{-\beta \gamma^2},
\]

where \( \gamma \) is some user-defined positive number.

A corollary of the above two results is the following.

**Corollary 1 Global Optimality.** Let \( \tilde{J}_1^t \) and \( \tilde{J}_2^t \) be the nominal costs for two different nominal trajectories with \( \tilde{J}_1^t < \tilde{J}_2^t \) for all \( t \). Then, there exists a sufficiently small \( \epsilon \), such that \( \tilde{J}_1^t(x_t^\epsilon,1) < \tilde{J}_2^t(x_t^\epsilon,2) \), given any arbitrarily high probability \( 1 - \delta \).

Thus, if the nominal plan is a global optimum plan, then the feedback plan consisting of the nominal, and the linear feedback plan associated with it, is also globally optimal.

**Decoupling.** Due to the decoupling result, the \( P_k \) equations which determine the feedback gains, do not affect the open loop (\( G_k \)) equations for an optimum nominal plan. Therefore, the nominal open loop design can be done completely independently of the feedback design. This can be done using any standard NLP solver in a blackbox fashion. Furthermore, the feedback design governed by the \( P_k \) equations is a standard Ricatti equation, and can be solved given the \( A_k, B_k, L_k \) functions. However, these are rather straightforward to estimate given the nominal trajectory, using random rollouts of the perturbed optimal system. Therefore, the decoupling result breaks the feedback law design into two "simpler" decoupled problems of open loop and linear feedback design. Finally, but not the least, it suggests a near optimal (to the third order) parametrization of the feedback law: an open loop design can be done completely independently of the feedback design. This can be done using any standard NLP solver in a blackbox fashion. Further more, the feedback design governed by the \( P_k \) equations is a standard Ricatti equation, and can be solved given the \( A_k, B_k, L_k \) functions. However, these are rather straightforward to estimate given the nominal trajectory, using random rollouts of the perturbed optimal system. Therefore, the decoupling result breaks the feedback law design into two "simpler" decoupled problems of open loop and linear feedback design. Finally, but not the least, it suggests a near optimal (to the third order) parametrization of the feedback law: an open loop control sequence + a linear feedback law wrapped around it, \( u_k(x_k^i) = \bar{u}_k + K_k \delta x_k^i \). The \( G_k \) are the Lagrange multipliers/ co-states in the problem and Eq. 2 corresponds to the first order optimality conditions for the nominal control problem.

**ILQG/DDP.** The condition in Eq. 2 is precisely when the ILQG/ DDP algorithms are deemed to have converged. However, that does not imply that the feedback gain in Eq. 3 has converged. In fact, in ILQG/ DDP, once the open loop has converged, the Ricatti equation in Eq. 3 would still need to be iterated in a policy iteration fashion till convergence to get the optimal feedback gain with respect to the optimized nominal trajectory. Again, this is evidence that the open loop and feedback problems are indeed decoupled.

**Large Deviations.** The large deviations bound in Eq. 4 has two parameters \( \gamma \) and \( \epsilon \) to control the accuracy of the estimate and its probability of validity. Using a suitable \( \gamma \), we can make the probability of violation (large deviation) as small as we desire. Similarly, by suitably choosing \( \epsilon \), we can make the error bound as small as we desire. Thus, this result shows that the error in the approximation \(|\hat{J}_k(.) - J_k(.)|\) incurred by only keeping the linear feedback term is \( O(\epsilon^3) \) with an arbitrarily high probability.

## 5 Decoupled Data Based Control (D2C) Design

In this section, we propose a novel decoupled data based control (D2C) approach. First, we solve a noiseless open-loop optimization problem to find a nominal optimal trajectory and then design a linearized closed-loop controller around the nominal trajectory, such that, with existence of stochastic perturbations, the state stays close to the optimal open-loop trajectory. The three-step framework to solve the stochastic feedback control problem may be summarized as follows.

- Solve the open loop optimization problem using a general nonlinear programming (NLP) solver with a black box simulation model of the dynamics.
- Linearize the system around the nominal open loop optimal trajectory, and identify the linearized time-varying system from input-output experiment data using a suitable system identification algorithm.
• Design an LQG controller which results in an optimal linear control policy around the nominal trajectory.

In the following section, we discuss each of the above steps.

5.1 Open Loop Trajectory Optimization

Consider the noiseless nonlinear system:

\[ x_{k+1} = f(x_k) + B_k u_k, \]

with known initial state \( x_0 \). The open loop state optimization problem given an initial state \( x_0 \) is:

\[ \{ u_k \}_{k=0}^{N-1} = \arg \min \sum_{k=0}^{N} (x_k - \tilde{x}_k)^2, \]

subject to the noiseless dynamics (Eq. 5). The open loop optimization problem is solved using a general NLP solver, where the underlying dynamic model is used as a blackbox, and the necessary gradients and hessians are found by the solver typically using finite differencing, which is also highly amenable to parallelization. For example, open loop optimization using gradient descent [10] is summarized in Appendix C.

5.2 Linear Time-Varying System Identification

Denote the optimal open-loop control as \( \{ \hat{u}_k \}_{k=0}^{N-1} \), and the corresponding nominal state as \( \{ \tilde{x}_k \}_{k=0}^{N-1} \).

We linearize the system around the nominal trajectory \( \{ \tilde{x}_k \} \), assuming that the control and disturbance enter through the same channels and the noise is purely additive (these assumptions are only for simplicity and can be relaxed easily):

\[ \delta x_{k+1} = A_k \delta x_k + B_k (\delta u_k + w_k), \]

where \( \delta x_k = x_k - \tilde{x}_k \) describes the state deviations from the nominal trajectory, \( \delta u_k = u_k - \tilde{u}_k \) describes the control deviations, and \( A_k = \frac{\partial f(x,u,w)}{\partial x} \bigg|_{\tilde{x}_k, \tilde{u}_k, 0}, B_k = \frac{\partial f(x,u,w)}{\partial u} \bigg|_{\tilde{x}_k, \tilde{u}_k, 0} \).

Consider system with zero noise and \( \delta x_0 = 0 \), the input-output relationship is given by:

\[ \delta x_k = \sum_{j=0}^{k-1} h_{k,j} \delta u_j, \]

where \( h_{k,j} \) is a generalized Markov parameters, and is defined by: \( h_{k,j} = A_{k-1} A_{k-2} \cdots A_{j+1} B_j \).

Partial Realization Problem [3 4]: Given a finite sequence of Markov parameters \( h_{k,j} \in \mathbb{R}^{n_r \times n_s}, k = 1, 2, \ldots, s, j = 0, 1, \ldots, k \), the partial realization problem consists of finding a positive integer \( n_r \) and LTV system \( (A_k, B_k, C_k) \), where \( A_k \in \mathbb{R}^{n_r \times n_r}, B_k \in \mathbb{R}^{n_r \times n_s}, C_k \in \mathbb{R}^{n_r \times n_r} \), such that the identified generalized Markov parameters \( h_{k,j} = C_k \tilde{A}_{k-1} A_{k-2} \cdots A_{j+1} B_j = h_{k,j} \). Then \( (A_k, B_k, C_k) \) is called a partial realization of the sequence \( h_{k,j} \).

We solve the partial realization problem using the time-varying ERA, and construct the identified deviation system:

\[ \delta a_{k+1} = \hat{A}_k \delta a_k + \hat{B}_k (\delta u_k + w_k), \]

where \( \delta a_k \in \mathbb{R}^{n_r} \) denotes the reduced order model (ROM) deviation states. Time-varying ERA starts by estimating the generalized Markov parameters via least squares, using input-output experiments consisting of random rollouts of the optimized nominal system, i.e., by simulating the system under control actions \( u_k = \tilde{u}_k + \omega_k \), where \( \tilde{u}_k \) is the nominal control action at time \( k \) and \( \omega_k \) is a random noise perturbation. Then, it constructs a generalized Hankel matrix, and solves the singular value decomposition (SVD) problem of the constructed Hankel matrix. The dimension \( n_r \) of the ROM is such that \( n_r \ll n_x \) when \( n_x \) is large, where \( n_x \) is the dimension of the state, thereby automatically providing a compact parametrization of the problem. The details of the time-varying ERA can be found in [16], and is briefly summarized in Appendix D.

5.3 Feedback Controller Design

Given the identified deviation system [8], we design the closed-loop controller to follow the optimal nominal trajectory, which is to minimize the quadratized cost function \( J_f = \sum_{k=0}^{N-1} (\delta a_k' Q_k \delta a_k + \delta u_k'^2 R_k \delta u_k). \)
\[ \delta u'_k R_k \delta u_k + \delta \dot{u}'_N Q_N \delta \dot{u}_N, \]

where \( \delta \dot{a}_k \) denotes the estimates of the deviation state \( \delta a_k, Q_k, Q_N \) are positive definite, and \( R_k \) is positive semi-definite. For the linear system (8), the "separation principle" of linear control theory (not the Decoupling result of Section 4) can be used [6]. Using this result, the design of the optimal linear stochastic controller can be separated into the independent design of an optimal Kalman filter and a fully observed optimal LQR controller. The details of the design is standard [6] and is omitted here.

The Decoupled Data Based Control Algorithm is summarized in Algorithm 1.

Algorithm 1 Decoupled Data Based Control Algorithm

1: Solve the deterministic open-loop optimization problem for optimal open loop nominal trajectory \( \{u_k\}_{k=0}^{N-1}, \{x_k\}_{k=0}^N \) using a general NLP solver (Section 5.1).
2: Identify the LTV system \( (\hat{A}_k, \hat{B}_k, \hat{C}_k) \) via time-varying ERA (Section 5.2).
3: Solve the decoupled Riccati equations using LTV system for feedback gain \( \{L_k\}_{k=0}^N \).
4: Set \( k = 0 \), given initial estimates \( \delta \hat{a}_0 = 0, P_0 \).
5: while \( k \leq N - 1 \) do
6: \[ u_k = \bar{u}_k - L_k \delta \hat{a}_k, \]
\[ x_{k+1} = f(x_k, u_k, w_k), \tag{9} \]
7: Update \( \delta \hat{a}_k \) using Kalman Filter.
8: \( k = k + 1 \).

5.4 Discussion

Complexity: The model free open loop optimization problem has complexity \( O(n_u) \), where \( n_u \) is the number of inputs, the LTV system identification step is again \( O(n_uy) \), and the LQG feedback design has complexity \( O(n_r^2) \), where \( n_r \) is the order of the ROM from the LTV system identification step. Suppose we were to use an ILQG based design such as in [2, 22], the complexity of the controller/policy parametrisation is \( O(n_u n_x^2) \). Moreover, the policy evaluation step would require the estimation of a parameter of the size \( O(n_x^4) \). For POMDPs, since \( n_r \ll n_x \) typically, the complexity of D2C approach is several orders of magnitude smaller.

Optimality: The open loop law generated by the NLP solver can be guaranteed to be locally optimal under usual regularity conditions. Theorems 1 and 2 show that the decoupled law is \( O(\epsilon^3) \) optimal, and therefore, shows robust behavior even with moderate to high levels of noise (please see Section 6).

6 Benchmark Examples

In this section, we illustrate the D2C approach using three benchmark examples from the RL literature, with comparison to the model based iLQG approach. We also compare the data efficiency of the proposed approach with respect to reported results in the RL literature.

The noise-to-signal ratio (NSR) is defined as: \( NSR = \sqrt{\frac{W}{\|u\|^2}} \), where \( W \) is the process noise covariance, and \( u \) is the control input.

We test the D2C approach using three benchmark fully observed examples: Cart-pole, Cart-two-pole (15) and Acrobot (19). The models are shown in Fig. 1. For Cart-pole and Cart-two-pole, the control is the force applied to the cart. For Acrobot, the torque is applied to the leg. For comparison, we use iLQG (20), where we assume that dynamic models of the examples required by iLQG are known (unknown to D2C).
6.1 Cart-Pole
The control objective is to swing up within 3 seconds, and keep balance between $[3, 3.5]s$. The initial states are $[x, \theta, \dot{x}, \dot{\theta}] = [0, \pi/4, 0, 0]$, and the target states are $[0, 0, 0, 0]$. The parameters are $M = 1.0, m = 0.1, L = 0.5$.

6.2 Cart-Two-Pole
The control objective is to swing up within 3 seconds. The initial states are $[x, \theta_1, \theta_2, \dot{x}, \dot{\theta}_1, \dot{\theta}_2] = [0, \pi/4, \pi/4, 0, 0, 0]$, and the target states are $[0, 0, 0, 0, 0, 0]$. The parameters are $M = 0.15, m_1 = 0.6, m_2 = 0.5, L_1 = 0.6, L_2 = 0.5$.

6.3 Acrobot
The control objective is to swing up within 5 seconds. The initial states are $[\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2] = [\pi/2, \pi/2, 0, 0]$, and the target states are $[0, 0, 0, 0]$. The parameters are $m_1 = m_2 = 1.0, l_1 = l_2 = 0.5, \mu_1 = \mu_2 = 0.05$, where $\mu_1, \mu_2$ are the friction coefficient of body and legs respectively.

The averaged computational time using D2C approach is shown in Table 1. The computational complexity comparison between D2C and iLQG is shown in Table 2. The D2C open loop optimization problem is solved using Matlab NLP solver fmincon, with an interior-point algorithm. The iLQG approach is implemented utilizing the Matlab toolbox provided in [20].

Table 1: Averaged Computational Time (s)

|                  | Open Loop | LTV ID | LQR | KF | Total Time |
|------------------|-----------|--------|-----|----|------------|
| Cart-Pole        | 22.5      | 4.79   | 0.01| 0.38| 27.68      |
| Cart-Two-Pole    | 52.69     | 4.97   | 0.01| 0.36| 57.67      |
| Acrobot          | 309.82    | 8.26   | 0.015| 0.41| 318.5      |

Table 2: Computational Complexity Comparison

|                  | Cart-Pole | Cart-Two-Pole | Acrobot |
|------------------|-----------|---------------|---------|
|                  | D2C       | iLQG          | D2C     | iLQG     | D2C      | iLQG     |
| # Iterations     | 255       | 582           | 106     | 131      | 160      | 148      |
| # Learning Trials| $2.34 \times 10^4$ | $3.35 \times 10^4$ | $1.6 \times 10^5$ | $\times$ | $\times$ | $\times$ |
| Off-line Time    | 27.3(s)   | 318.25(s)     | 57.67(s)| 816(s)   | 318.1(s) | 442.27(s) |
| On-line Time     | 0.38(s)   | 0.36(s)       | $\times$ | $\times$ | 0.41(s)  | $\times$  |
| Total Time       | 27.68(s)  | 318.25(s)     | 58.03(s)| 816(s)   | 318.5(s) | 442.27(s) |

In all examples, the off-line computational time using D2C consists of the open loop optimization, LTV system identification and LQR design shown in Table 1. The learning trials using D2C approach is the number of system rollouts used in the open loop optimization (22000, 32200 and $1.6 \times 10^5$) and the LTV system identification (1372, 1324 and 565). Starting from the same initial guess, D2C and iLQG converge to the optimal solution with increasing number of iterations. In Fig. 2 the comparison of the convergence rate after the 1st iteration between D2C and iLQG for
Cart-Pole balance and Cart-two-Pole swing up and Acrobot swing up are shown. The performance of the Cart-Pole swing up using both approaches are almost the same and are omitted here. We run 100 Monte-Carlo simulations, for Cart-pole and Cart-two-Pole, the comparison of the averaged cost using D2C and iLQG as a function of the NSR is shown in Fig. 3. For Acrobot example, the averaged performance using D2C and iLQG when $NSR = 0.01$ are plotted in Fig 4. In all examples, the same cost functions and initial guess are used, and are chosen “fairly” such that both approaches could converge to a good solution.

![Comparison of the Convergence Rate](image)

Figure 2: Comparison of the Convergence Rate.

![Comparison of the averaged cost](image)

Figure 3: Comparison of the averaged cost.

![Performance Comparison for Acrobot](image)

Figure 4: Performance Comparison for Acrobot

It can be seen that 1) the computational time using D2C is much smaller than iLQG due to the decoupling as we do not have to solve the backward Ricatti equation at every iteration, (which takes up bulk of the time), as required by iLQG. The online computation using D2C is around 0.4s; 2) the performance of D2C approach is comparable to the iLQG approach. Further, the D2C approach seems less sensitive to the process noise when compared to the iLQG approach; and 3) we assume the dynamics is given to the iLQG approach, which is unavailable to the D2C approach.
6.4 DeepRL comparison

We note that for the tasks performed in this section, Deep RL algorithms such as the DDPG typically need $O(10^6) - O(10^8)$ learning trials, and several hours of learning time, even with parallelization. In contrast, we see that the D2C approach is able to accomplish the same tasks using $10^3 - 10^5$ learning trials, and in several minutes on a laptop computer. This, in our opinion, shows the data efficiency of the proposed D2C approach for continuous control problems [5, 15, 19]. However, this should not be surprising since the D2C approach is more efficient than a model based approach such as iLQG which is used as a benchmark by the Deep RL techniques [19].

7 Conclusion

In this paper, we have proposed a decoupled data based design of the stochastic optimal control problem for systems with unknown nonlinear dynamics. First, we design a deterministic open-loop optimal trajectory. Then we identify the nominal linearized system using time-varying ERA. The open-loop optimization and system identification are implemented offline, using the random perturbations/impulse responses of the system, and an LQG controller based on the ROM is implemented online. We have tested the proposed approach on several fully observed benchmark examples, and showed the performance of the proposed approach with respect to the model based iLQG approach as well as state of the art RL techniques such as DDPG. Future work will generalize the proposed approach to large-scale partially observed systems.

References

[1] A.E.Bryson and W.F. Denham. A steepest-ascent method for solving optimum programming problems. *Journal of Applied Mechanics*, 29(2), 1962.

[2] R. Akrou, A. Abdolmaleki, H. Abdulsamad, and G. Neumann. Model Free Trajectory Optimization for Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning*, 2016.

[3] A.M. King, U.B. Desai, and R.E. Skelton. A Generalized Approach to q-Markov Covariance Equivalent Realizations for Discrete Systems. *Automatica*, 24(4):507–515, 1988.

[4] A.C. Antoulas. *Approximation of Large Scale Dynamical Systems*. SIAM, Philadelphia, PA, 2005.

[5] G Barth-Maron et al. Distributed distributional deterministic policy gradients. In *Proc. ICLR*, 2018.

[6] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control, Two Volume Set*. Athena Scientific, 2nd edition, 1995.

[7] Robert P. Bithmead, Vincent Wertz, and Michael Gerers. *Adaptive Optimal Control: The Thinking Man’s G.P.C*. Prentice Hall Professional Technical Reference, 1991.

[8] Maurizio Falcone. Recent Results in the Approximation of Nonlinear Optimal Control Problems. In *Large-Scale Scientific Computing LSSC*, 2013.

[9] Abhijit Gosavi. *Simulation-based optimization: Parametric optimization techniques and reinforcement learning*. Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[10] D. Jacobsen and D. Mayne. *Differential Dynamic Programming*. Elsevier, 1970.

[11] Said G. Khan et al. Reinforcement learning and optimal adaptive control: An overview and implementation examples. *Annual Reviews in Control*, 36:42–59, 2012.

[12] S. Levine and P. Abbeel. Learning Neural Network Policies with Guided Search under Unknown Dynamics. In *Advances in Neural Information Processing Systems*, 2014.

[13] S. Levine and K. Vladlen. Learning Complex Neural Network Policies with Trajectory Optimization. In *Proceedings of the International Conference on Machine Learning*, 2014.

[14] W. Li and E. Todorov. Iterative linearization methods for approximately optimal control and estimation of non-linear stochastic system. *International Journal of Control*, 80(9):1439–1453, 2007.
[15] T Lillicrap et al. Continuous control with deep reinforcement learning. In Proc. ICLR, 2016.

[16] Manoranjan Majji, Jer-Nan Juang, and John L. Junkins. Time-varying Eigensystem Realization Algorithm. Journal of Guidance, Control, and Dynamics, 33(1):13–28, 2010.

[17] Djordje Mitrovic, Stefan Klanke, and Sethu Vijayakumar. Adaptive Optimal Feedback Control with Learned Internal Dynamics Models, pages 65–84. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[18] D Silver et al. Deterministic policy gradient algorithms. In Proc. ICML, 2014.

[19] Y. Tassa et al. Deepmind control suite. arXiv preprint arXiv:1801.00690v1, 2018.

[20] Yuval Tassa, N. Mansard, and E. Todorov. Control-Limited Differential Dynamic Programming. In Robotics and Automation (ICRA), 2014 IEEE International Conference on, 2014.

[21] Evangelos Theoddorou, Yuval Tassa, and Erno Todorov. Stochastic Differential Dynamic Programming. In Proceedings of American Control Conference, 2010.

[22] E. Todorov and Y. Tassa. Iterative Local Dynamic Programming. In Proc. of the IEEE Int. Symposium on ADP and RL., 2009.

[23] Emanuel Todorov and Weiwei Li. A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In Proceedings of American Control Conference, pages 300 – 306, 2005.

[24] Xiangnan Zhong, Haibo He, Huaguang Zhang, and Zhanshan Wang. Optimal Control for Unknown Discrete-Time Nonlinear Markov Jump Systems Using Adaptive Dynamic Programming. IEEE Transactions on Neural networks and learning systems, 25(12):2141–2155, 2014.
A Decoupled Data Based Control Approach to Stochastic Optimal Control Problems Supplementary Material

Dan Yu∗
College of Astronautics
Nanjing University of Aeronautics and Astronautics
Nanjing, China, 210016
yudan198811@hotmail.com

Mohammandhussen Rafieisakhaei
Department of Electrical and Computer Engineering
Texas A&M University
College Station, TX, 77840
mrafieis@tamu.edu

Suman Chakravorty
Department of Aerospace Engineering
Texas A&M University
College Station, TX, 77840
schakrav@tamu.edu

This supplementary material is organized as follows. We prove Theorem 1 and 2 in Appendices A and B. In Appendix C we present the open loop optimization approach. In Appendix D we review the time-varying eigensystem realization algorithm (ERA).

A Proof of Theorem 1

Consider the Dynamic Programming equation corresponding to the true stochastic control problem:

\[
J_k(x_k^*) = \min_{u_k} \{c(x_k^*, u_k) + E[J_{k+1}(x_{k+1}^*)]\},
\]

(1)

where the Q-function is given by:

\[
Q_k(x_k^*) = c(x_k^*, u_k) + E[J_{k+1}(x_{k+1}^*)],
\]

(2)

and the terminal cost function is \(J_N(x_N^*) = \phi_N(x_N^*)\). Recall that \(J_k(x_k^*) = \bar{J}_k + G_k \delta x_k^* + \delta x_k^* P_k \delta x_k^* + \Phi_k(\delta x_k^*)\). Then, it follows that: \(J_{k+1}(x_{k+1}^*) = \bar{J}_{k+1} + G_{k+1} \delta x_{k+1} + \delta x_{k+1} P_{k+1} \delta x_{k+1} + \Phi_{k+1}(\delta x_{k+1})\), where \(\delta x_{k+1} = A_k \delta x_k^* + B_k \delta u_k + \epsilon B_{k+1} w_k + r_k\). Thus:

\[
E[J_{k+1}(x_{k+1}^*)] = \bar{J}_{k+1} + G_{k+1} A_k \delta x_k^* + G_{k+1} B_k \delta u_k
\]

\[
+ G_{k+1} c_k + \delta x_k^* A_k' P_{k+1} A_k \delta x_k^* + \delta u_k B_k' P_{k+1} A_k \delta x_k^*
\]

\[
+ \delta x_k^* A_k' P_{k+1} B_k \delta u_k + \delta u_k B_k' P_{k+1} B_k \delta u_N
\]

\[
+ \epsilon^2 \text{tr}(P_k^{1/2} B_k' B_k' P_k^{1/2}) + \delta u_k B_k' P_{k+1} B_k \Phi_{k+1}(\delta u_k)
\]

∗Use footnote for providing further information about author (webpage, alternative address) — not for acknowledging funding agencies.
We may now use (3) and (4) to do the minimization of the Q-function on the RHS of (1). Using the \( \tilde{S} \) function. Therefore, it follows that for sufficiently small \( \delta x_k \),

\[
\frac{\partial Q}{\partial u} = 0
\]

We may now use (3) and (4) to do the minimization of the Q-function on the RHS of (1). Using the fact that \( \min_{u_k} Q(x_k, u_k) = \min_{\delta x_k} Q(\delta x_k, \delta u_k) \), and setting \( \frac{\partial Q}{\partial u_k} = 0 \), we obtain:

\[
\delta u^*_k = -S^{-1}_k(R_k \bar{u}_k + G_{k+1} B_k) - S^{-1}_k(2B_k' P_{k+1} A_k) \delta x^*_k
\]

\[
\frac{\partial}{\partial u_k} \frac{\partial Q}{\partial u} = 0
\]

where \( S_k = \frac{1}{2} R_k + B_k^T P_{k+1} B_k \). If \( \bar{u}_k \) is an optimal control sequence, then it follows from the first order optimality conditions that \( R_k \bar{u}_k + G_{k+1} B_k = 0 \) for all \( k \). Utilize this fact, substitute Eq. (5) back into (2) to get the optimal \( Q_k(\delta x^*_k, \delta u^*_k) \), and then substitute the resulting \( Q_k(\delta x^*_k, \delta u^*_k) \) back into the DP (1). Noting that \( -\frac{1}{2} S_k K_k = B_k^T P_{k+1} A_k \), and grouping the different powers of \( \delta x^*_k \) on both sides of the resulting DP equation results in the following equations:

\[
G_k = L_k + R_k \bar{u}_k K_k + G_{k+1} (A_k + B_k K_k),
\]

\[
P_k = L_{kk} + \frac{1}{2} K'_k R_k K'_k + A'_k P_{k+1} A_k - K'_k S_k K_k + K'_k B_k^T P_{k+1} B_k K_k + G_{k+1} \bar{R}_{k,xx},
\]

where \( \bar{R}_{k,xx} = \nabla^2 f(x_k) \) is the second order terms in the residual dynamics term \( r_k(\delta x^*_k) \). Regrouping the \( G_k \) equation, and noting that \( \bar{u}_t \) is a nominally optimal sequence gives:

\[
G_k = L_k + (R_k \bar{u}_k + G_{k+1} B_k) K_k + G_{k+1} A_k
\]

\[
= L_k + G_{k+1} A_k.
\]

This completes the first part of our assertion.

Noting that \( |J_k(x_k^*) - \bar{J}_k(x_k^*)| = |q_k(\delta x^*_k)| \), it follows that the error incurred is \( O(||\delta x^*_k||) \) since \( q_k \) is the same. This completes the proof.

**B Proof of Theorem 2**

First, we state the following lemma for a large deviation bound on a linear Gaussian random process. The proof is relatively straightforward but tedious and omitted here.

**Lemma 1** Consider the linear dynamical system \( x_{k+1} = A_k x_k + \epsilon B_k w_k \), where \( w_k \) is a white noise process. There exist constants \( \alpha, \beta \), only dependent on the system matrices \( A_k, B_k \) such that:

\[
\text{Prob}( \max_{0 \leq k \leq N} ||\delta x_k|| > M) \leq \alpha \frac{e^{-\frac{\beta M^2}{2}}}{M}
\]

Let \( \delta x_k \) denote the solution of the fictitious linear system \( \delta x_k = \bar{A}_k \delta x_k + \epsilon B_k w_k \), where \( \bar{A}_k = A_k + \bar{B}_k K_k \), and \( K_k \) is the feedback gain matrix calculated from Theorem 1.

**Proof of Theorem 2.**

Let \( \text{Prob}(\max_{0 \leq k \leq N} ||\delta x_k|| \geq M) \leq \delta \). It can be shown that \( ||\delta x_k - \bar{x}_k|| \) is an \( O(\max_{0 \leq k \leq N} ||\delta x_k||) \) function. Therefore, it follows that for sufficiently small \( \delta x_k \): \( \text{Prob}(\max_{0 \leq k \leq N} ||\delta x_k|| \geq M) \leq \delta \). Utilizing Assumption 1, we have that \( |J_k(\delta x_k) - \bar{J}_k(\delta x_k)| \leq L ||\delta x_k||^3 \), for sufficiently small \( ||\delta x_k|| \). Thus, it follows that \( \text{Prob}(\max_{0 \leq k \leq N} |J_k(\delta x_k) - \bar{J}_k(\delta x_k)| \geq L M^3) \leq \delta \).
The open loop state optimization problem given an initial state \( x_0 \) is solved using an NLP solver, where the underlying dynamic models are not needed.

The control policy is updated iteratively via gradient descent, where the gradient vector is defined as:

\[
\nabla \bar{J}(x_n, u_n) = \left( \frac{\partial \bar{J}}{\partial x_0}, \frac{\partial \bar{J}}{\partial u_1}, \ldots, \frac{\partial \bar{J}}{\partial u_{N-1}} \right) |_{x_n, u_n}
\]

and without knowing the explicit form of the cost function, each partial derivative with respect to the \( i^{th} \) control variable \( u_i \) is calculated as follows:

\[
\frac{\partial \bar{J}}{\partial u_i} |_{x_n, u_n} = \frac{\bar{J}(x_1^{(n)}, u_0^{(n)}, \ldots, u_i^{(n)} + h, \ldots, u_{N-1}^{(n)}) - \bar{J}(x_1^{(n)}, u_0^{(n)}, \ldots, u_i^{(n)}, \ldots, u_{N-1}^{(n)})}{h}
\]

where \( h \) is a small constant perturbation and \( x_i^{(n)} \) denotes the state corresponding to the control input \( \{u_0^{(n)}, \ldots, u_i^{(n)} + h, \ldots, u_{N-1}^{(n)}\} \), \( i = 0, \ldots, N - 1 \).

The gradient descent procedure is summarized in Algorithm 1.

### C Open Loop Optimization Algorithm

The open loop state optimization problem given an initial state \( x_0 \) is:

\[
\{u_k\}_{k=0}^{N-1} = \arg \min_{\{u_k\}} \bar{J}(\{x_k\}_{k=0}^{N}, \{u_k\}_{k=0}^{N-1}),
\]

\[
x_{k+1} = f(x_k, u_k, 0), y_k = x_k.
\]

The problem is solved using an NLP solver, where the underlying dynamic models are not needed. For example, open loop optimization using gradient descent is summarized as follows.

Denote the initial guess of the control sequence as \( U^{(0)} = \{u_k^{(0)}\}_{k=0}^{N-1} \), and the corresponding states \( x^{(0)} = \{x_k^{(0)}\}_{k=0}^{N} \).

The control policy is updated iteratively via

\[
U^{(n+1)} = U^{(n)} - \alpha \nabla U \bar{J}(x^{(n)}, U^{(n)}),
\]

until a convergence criterion is met, where \( U^{(n)} = \{u_k^{(n)}\}_{k=0}^{N-1} \) denotes the control sequence in the \( n^{th} \) iteration, \( x^{(n)} = \{x_k^{(n)}\}_{k=0}^{N} \) denotes the corresponding states, and \( \alpha \) is the step size parameter.

### D Time-Varying Eigensystem Realization Algorithm

Consider the linear time-varying (LTV) system:

\[
\begin{align*}
\delta x_{k+1} &= A_k \delta x_k + B_k \delta u_k + G_k w_k, \\
\delta y_k &= C_k \delta x_k + F_k w_k,
\end{align*}
\]

where \( \delta x_k = x_k - \bar{x}_k \) describes the state deviations from the nominal trajectory, \( \delta u_k = u_k - \bar{u}_k \) describes the control deviations, \( \delta y_k = y_k - h(\bar{x}_k, 0) \) describes the measurement deviations, and \( A_k \in \mathbb{R}^{n_x \times n_x}, B_k \in \mathbb{R}^{n_x \times n_u}, C_k \in \mathbb{R}^{n_y \times n_x} \).
Algorithm 1 Gradient Descent Algorithm

Require: Initial state $x_0$, cost function $J()$, initial guess $U^{(0)} = \{u_k^{(0)}\}_{k=0}^{N-1}$, gradient descent design parameters $\alpha, h, \epsilon$.

Ensure: Optimal control sequence $\{\bar{u}_k\}_{k=0}^{N-1}$, nominal trajectory $\{\bar{x}_k\}_{k=0}^N$

1: $n = 0$, set $\nabla U \tilde{J}(X^{(0)}, U^{(0)}) = \epsilon$.
2: while $\nabla U \tilde{J}(X^{(n)}, U^{(n)}) \geq \epsilon$ do
3: Evaluate the cost function $\tilde{J}(X^{(n)}, U^{(n)})$
4: Perturb each control variable $u_i^{(n)}$ by $h$ and compute the state $X_i^{(n)}$, $i = 0, \cdots, N - 1$.
5: Update the control policy $U^{(n+1)} = U^{(n)} - \alpha \nabla U \tilde{J}(X^{(n)}, U^{(n)})$.
6: $n = n + 1$.
7: end while
8: $\{u_k\}_{k=0}^{N-1} = U^{(n)}$, $\{\bar{x}_k\}_{k=0}^N = f(x_0, U^{(n)}, 0)$.

Define the generalized Markov parameters $h_{k,j}$ as:

$$h_{k,j} = \begin{cases} C_k A_{k-1} A_{k-2} \cdots A_{j+1} B_j, & \text{if } j < k - 1, \\
C_k B_{k-1}, & \text{if } j = k - 1, \\
0, & \text{if } j > k - 1,
\end{cases}$$

(13)

and define the generalized Hankel matrix as:

$$H_{k}^{(p,q)} = \begin{pmatrix} h_{k,k-1} & h_{k,k-2} & \cdots & h_{k,k-q} \\
h_{k+1,k-1} & h_{k+1,k-2} & \cdots & h_{k+1,k-q} \\
\vdots & \vdots & \ddots & \vdots \\
h_{k+p-1,k-1} & h_{k+p-1,k-2} & \cdots & h_{k+p-1,k-q}\end{pmatrix},$$

(14)

where $p$ and $q$ are design parameters could be tuned for best performance, and $pn_y \geq n_r, qn_u \geq n_r$, $n_r$ is the rank of the Hankel matrix.

Given the generalized Markov parameters, we construct two Hankel matrices $H_{k}^{(p,q)}$ and $H_{k+1}^{(p,q)}$, and then solve the singular value decomposition problem:

$$H_{k}^{(p,q)} = U_k \Sigma_k^{1/2} \Sigma_k'^{1/2} V_k^\prime,$$

(15)

where $(.)'$ denotes the transpose of $(.)$. Denote the rank of the Hankel matrix $H_{k}^{(p,q)}$ is $n_r$, where $n_r \leq n_r$. Then $\Sigma_k \in \mathbb{R}^{n_r \times n_r}$ is the collection of all non-zero singular values, and $U_k \in \mathbb{R}^{pn_y \times n_r}$, $V_k \in \mathbb{R}^{pn_u \times n_r}$ are the corresponding left and right singular vectors.

Similarly, $H_{k+1}^{(p,q)} = O_{k+1}^{(p)} R_{k}^{(q)}$.

The identified system using time-varying ERA is:

$$\hat{A}_k = (O_{k+1}^{(p)})^+ O_k^{(p)},$$

$$\hat{B}_k = R_{k}^{(q)}(:, 1 : n_u),$$

$$\hat{C}_k = O_k^{(p)}(1 : n_y, :)$$

(16)

where $(.)^+$ denotes the pseudo inverse of $(.)$, $O_{k+1}^{(p)}$ contains the first $(p - 1)n_y$ rows of $O_k^{(p)}$, and $O_k^{(p)}$ contains the last $(p - 1)n_y$ rows of $O_k^{(p)}$. Here, we assume that $n_r$ is constant through the time period of interest, which could also be relaxed.
We run \( M \) simulations and in the \( i \)th simulation, choose input sequence \( \{\delta u_k(i)\}_{k=0}^{N-1} \), and collect the output \( \delta y_k(i) \). The subscript \( (i) \) denotes the experiment number. Then the generalized Markov parameters \( \{h_{k,j}\}_{j=0}^{k-1} \) could be recovered via solving the least squares problem:

\[
\begin{pmatrix}
\delta y_{k,1} & \delta y_{k,2} & \cdots & \delta y_{k,M}
\end{pmatrix} = \begin{pmatrix}
h_{k,0} & h_{k,1} & \cdots & h_{k,N-1}
\end{pmatrix} \begin{pmatrix}
\delta u_{k,0} & \delta u_{k,1} & \cdots & \delta u_{k,N-1}
\end{pmatrix},
\tag{18}
\]

where \( M \) is a design parameter and is chosen such that the least squares solution is possible.

Notice that we cannot perturb the system (12) directly. Instead, we identify the generalized Markov parameters as follows.

Run \( M \) parallel simulations with the noise-free system:

\[
x_{k+1,(i)} = f(x_{k,(i)}, \bar{u}_k + \delta u_{k,(i)}, 0), \\
y_{k,(i)} = h(x_{k,(i)}, 0),
\tag{19}
\]

where \( i = 1, 2, \cdots, M \), and therefore,

\[
\delta y_{k,(i)} = y_{k,(i)} - h(\bar{x}_k, 0).
\tag{20}
\]

where \((\bar{u}_k, \bar{x}_k)\) is the open loop optimal trajectory. Then we solve the same least squares problem with (18).

For simplicity, we assume that the process noise is added through controllers, and the measurement noise is independent of the state and control variables, i.e., \( G_k = B_k, F_k = I_{n_y \times n_y} \), while the proposed algorithm is extendable to identify \( G_k \) and \( F_k \).

In general, the identified deviation system is:

\[
\begin{align*}
\delta a_{k+1} &= \hat{A}_k \delta a_k + \hat{B}_k \delta u_k + \hat{G}_k w_k, \\
\delta y_k &= \hat{C}_k \delta a_k + \hat{F}_k v_k,
\end{align*}
\tag{21}
\]

where \( \delta a_k \in \mathbb{R}^{n_x} \) denotes the reduced order deviation states.

The time-varying ERA used in this paper is summarized in Algorithm 2.