Accelerating Model-Free Policy Optimization Using Model-Based Gradient: A Composite Optimization Perspective

Yansong Li YLI340@UIC.EDU
Shuo Han HANSHUO@UIC.EDU
University of Illinois at Chicago, Chicago, IL 60607, USA

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

We develop an algorithm that combines model-based and model-free methods for solving a nonlinear optimal control problem with a quadratic cost in which the system model is given by a linear state-space model with a small additive nonlinear perturbation. We decompose the cost into a sum of two functions, one having an explicit form obtained from the approximate linear model, the other being a black-box model representing the unknown modeling error. The decomposition allows us to formulate the problem as a composite optimization problem. To solve the optimization problem, our algorithm performs gradient descent using the gradient obtained from the approximate linear model until backtracking line search fails, upon which the model-based gradient is compared with the exact gradient obtained from a model-free algorithm. The difference between the model gradient and the exact gradient is then used for compensating future gradient-based updates. Our algorithm is shown to decrease the number of function evaluations compared with traditional model-free methods both in theory and in practice.

Keywords: Model-based control, model-free control, composite optimization, gradient methods

1. Introduction

Learning-based algorithms have flourished in the field of control (Recht, 2019) in recent years with applications in robotic manipulation and locomotion (Kohl and Stone, 2004; Levine et al., 2016), energy systems (Chen et al., 2021), and transportation (Wu et al., 2017). Learning-based control algorithms can be classified into two categories: model-free algorithms and model-based algorithms. Model-free algorithms do not utilize an explicit form of the dynamics and can handle dynamics that are hard to model. In the meantime, model-free algorithms require a large amount of data to learn an optimal policy. This may become impractical for physical systems, in which data collection is often expensive since every collection involves physical interactions with the environment.

In contrast, model-based algorithms maintain an explicit form of the dynamics such as a state-space model. With the aid of a model, model-based algorithms usually require less amount of data (Tu and Recht, 2019) to learn an optimal policy. However, model-based algorithms tend to have difficulties in modeling complex dynamics and may suffer from model bias (Deisenroth et al., 2015; Atkeson and Santamaria, 1997; Abbeel et al., 2006) when the model class is not sufficiently rich.

In this paper, we develop a method that combines model-free and model-based methods to learn an optimal policy for controlling a nonlinear system under a quadratic cost. We formulate the optimal control problem as a composite optimization problem, in which the cost is expressed as a

composite function defined by the sum of a model-based part and a model-free part: the model-based part has an analytical form, whereas the model-free part is viewed as a black box. We then develop a hybrid gradient-based algorithm that searches for the optimal solution using the *model gradient*, i.e., the gradient of the model-based part, plus a corrective compensation term, where the compensation is intermittently updated by the gradient of the model-free part. The algorithm is shown to require fewer function evaluations compared with purely model-free algorithms both in theory and in practice. Due to space constraints, proofs in the paper are omitted can be found in the extended version (Li and Han, 2022).

Related work There have been several attempts in combining model-based and model-free algorithms. Qu et al. (2020) considered a modification of the linear quadratic regulator (LQR) problem, where a small additive nonlinear perturbation is introduced to the original linear dynamics. They first applied model-based control using the approximate model given by the linear part of the dynamics to obtain a near-optimal policy. Then, they showed that, using the near-optimal policy as the initial policy, the model-free policy gradient method is able to produce an optimal policy for the modified LQR problem. Chebotar et al. (2017) developed an algorithm to learn time-varying linear-Gaussian controllers for reinforcement learning, in which the policy update is decomposed into a model-based component and a model-free component. Shashua et al. (2021) used a descent algorithm with a gradient mapping that is updated in each iteration based on the model and interactions with the environment. Our algorithm is closest to the one introduced by Abbeel et al. (2006). They used inaccurate dynamics to train a near-optimal policy, which was used to collect trajectories from the real Markov decision process (MDP). For each state-action pair from the collected trajectories, they compared the next state in the trajectories with the one predicted by the inaccurate MDP dynamics and used the difference to update the inaccurate MDP dynamics by adding a bias term. In comparison, our algorithm focuses on the control cost rather than the system dynamics.

Our work is also related to policy gradient for LQR, discussed in Fazel et al. (2018) and Bu et al. (2019), in that we also use the same zeroth-order method to compute the gradient of the cost. See Conn et al. (2009) for a comprehensive coverage of zeroth-order methods in optimization.

Finally, our work is closely related to composite optimization. A classical method for solving composite optimization problems is the proximal gradient method (Rockafellar, 1976; Beck and Teboulle, 2009; Parikh and Boyd, 2014; Nesterov, 2013). The method assumes that the objective function can be expressed as the sum of a differentiable convex function and another possibly non-differentiable convex function. Moreover, the non-differentiable convex function is assumed to be associated with a proximal operator that can be efficiently evaluated. In contrast, our algorithm deals with composite optimization problems in which both functions in the sum are differentiable but does not assume the existence of an efficiently computable proximal operator.

2. Background: Nonlinear optimal control with quadratic cost

We consider a modified discrete-time LQR problem, which appeared in Qu et al. (2020). The dynamical system is described by a nonlinear state-space model with state $x_t \in \mathbb{R}^n$ and control input $u_t \in \mathbb{R}^p$,

$$x_{t+1} = Ax_t + Bu_t + h(x_t, u_t), \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times p}$ and $h\colon \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ with $h(0,0) = 0$. The nonlinear function h is not assumed to admit an explicit form but is considered "small," where the precise meaning

of "small" will be discussed in Section 4.3. The goal is to find a linear state feedback controller $u_t = -Kx_t$ (see footnote[1]) that minimizes the cost $C\colon \mathbb{R}^{p\times n} \to \mathbb{R}$ defined by

$$C(K) = \mathbb{E}_{x_0\sim\mathcal{D}}\left(\sum_{t=0}^{\infty} x_t^\top Q x_t + u_t^\top R u_t\right), \tag{2}$$

where $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{p\times p}$ are positive definite and x_0 is the initial state and is drawn from a fixed distribution $\mathcal{D}$. Multiple problems can be formulated as (1). For example, the dynamics of a robotic manipulator can be described by a linear state-space model plus an error term h. Note that if h is zero everywhere, i.e., $h \equiv 0$, the problem is the same as the vanilla LQR problem, which has a closed-form solution. We define $\hat{C}$ as the quadratic cost (2) when $h \equiv 0$ in (1). We denote the optimal point that minimizes $\hat{C}$ by $\hat{K}^\star$ and the optimal point that minimizes C by $K^\star$.

Throughout this paper, we propose to minimize C using a policy gradient method that uses gradient computed from a *zeroth-order method* (Fazel et al., 2018), which only requires access to the values of the cost rather than an analytical expression of the gradient mapping. The method is model-free since it does not rely on an explicit form of the system dynamics. Because each evaluation of the cost function requires collecting trajectories generated from the dynamical system, model-free policy gradient methods need to sample numerous trajectories in order to find a near-optimal policy (Tu and Recht, 2019). This issue of high sample complexity has been recognized as a major bottleneck in applying model-free policy gradient methods to physical systems, where the collection of trajectories are often expensive.

3. Proposed framework

To reduce the sample complexity of model-free policy gradient methods, we propose a policy optimization framework that leverages an inexact model of the system, given by the linear part of the dynamics. Our key idea is to reformulate the original problem as a composite optimization problem that explicitly shows how model information enters policy optimization. The revelation of the role of the model naturally leads to an optimization algorithm that solves the composite optimization problem with fewer function evaluations than traditional model-free policy gradient methods.

3.1. A composite optimization perspective of policy optimization

We decompose the cost function C into two components as

$$C(K) = \hat{C}(K) + r(K), \tag{3}$$

where r is the residual term defined by $r \triangleq C - \hat{C}$. The problem of minimizing the cost in (3) is an unconstrained optimization problem of the following form:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x), \tag{4}$$

where $f = \hat{f} + r$. We call $\nabla f(x)$ the *exact gradient* and $\nabla \hat{f}(x)$ the *model gradient* at x. Denote the optimal value of (4) by $f^\star$ and an optimal solution of (4) by $x^\star$. Also, denote $\hat{x}^\star$ as a point where $\nabla \hat{f}(\hat{x}^\star) = 0$. We make the following assumptions throughout the paper.

1. The goal is not to find an optimal state feedback controller, which may be nonlinear for nonlinear systems even when the cost is quadratic.

Assumption 1 *The function f satisfies the PL-condition, i.e., there exists some $\mu > 0$ such that $\|\nabla f(x)\|^2 \geq 2\mu(f(x) - f^\star)$ for all x.*

Assumption 2 *The residual mapping r is L_r-smooth, i.e., ∇r is L_r-Lipschitz continuous.*

Assumption 3 *The gradient value $\nabla f(x)$ can be computed with n function evaluations, where n is the dimension of x. The model gradient $\nabla \hat{f}$ has a closed-form expression.*

For the modified LQR problem defined by (1) and (2), Assumption 1 is satisfied when K is close to $K^\star$, since C is locally strongly convex near $K^\star$ (Qu et al., 2020), and strong convexity implies the PL-condition (Karimi et al., 2016). Assumption 2 is also satisfied when K is close to $K^\star$ due to the local smoothness of C (Qu et al., 2020). Assumption 3 is satisfied since $\nabla C(K)$ can be computed by zeroth-order methods, and $\nabla \hat{C}(K)$ has a closed-form solution (Fazel et al., 2018).

3.2. A model-exploiting composite optimization algorithm

Throughout this paper, we focus on descent methods for solving the problem in (4). At each iteration, the iterate is updated from x to x_+ by $x_+ = x - t\Delta(x)$, where t is the step size, and $\Delta(x)$ is a descent direction satisfying $f(x - t\Delta(x)) < f(x)$ when t is sufficiently small.

To solve the problem in (4), Qu et al. (2020) used the gradient methods with the exact gradient starting from $x = \hat{x}^\star$. The algorithm in Qu et al. (2020) can be viewed as applying the gradient methods with $\nabla \hat{f}$ until $\nabla \hat{f}(x) = 0$, i.e., $x = \hat{x}^\star$, after which the algorithm switches to using the exact gradient ∇f. In our algorithm, shown in Algorithm 1, instead of switching to ∇f permanently, we only evaluate ∇f intermittently at points $\tilde{x}^{(1)}, \tilde{x}^{(2)}, \ldots$ and use the evaluations to form *compensated model gradient mappings* $\tilde{g}^{(1)}, \tilde{g}^{(2)}, \ldots$ with $\tilde{g}^{(i)} \triangleq \nabla \hat{f} + \delta^{(i)}$, where $\delta^{(i)} \triangleq \nabla f(\tilde{x}^{(i)}) - \nabla \hat{f}(\tilde{x}^{(i)})$. By Assumption 3, $\delta^{(i)}$ can be computed with n function evaluations. The compensation $\delta^{(i)}$ remains unchanged until the algorithm reaches a point $\tilde{x}^{(i+1)}$ at which $\tilde{g}^{(i)}$ is no longer a descent direction and makes line search fail (LSF), after which a new compensation $\delta^{(i+1)}$ is computed by evaluating $\nabla f(\tilde{x}^{(i+1)})$. In the following, we say that the algorithm operates in the *model-free regime* when the exact gradient is used and in the *model-based regime* when the $\tilde{g}$ is used. The difference between our algorithm and the algorithm in Qu et al. (2020) is shown below:

- Qu et al. (2020): $\nabla \hat{f} \xrightarrow{\nabla \hat{f}=0} \nabla f$
- Our algorithm: $\nabla \hat{f} \xrightarrow{\nabla \hat{f}=0} \nabla f \to \tilde{g}^{(1)} \xrightarrow{\text{LSF}} \nabla f \to \tilde{g}^{(2)} \xrightarrow{\text{LSF}} \cdots$

Algorithm 1 Gradient compensation algorithm

Obtain η from backtracking line search (BLS) at x with the compensated model gradient $\tilde{g}(x)$.
if BLS succeeds at x **then** ▷ Model-based Regime
    $x \leftarrow x - \eta \tilde{g}(x)$
else ▷ Model-free Regime
    Apply BLS with the exact gradient $\nabla f(x)$ to obtain η.
    $x \leftarrow x - \eta \nabla f(x)$
    $\tilde{g} \leftarrow \nabla \hat{f} + \nabla f(x) - \nabla \hat{f}(x)$
end if

4. Main result

When ∇f is used for computing the update x_+, our algorithm is identical to the vanilla gradient descent. Therefore, we only need to consider the case when x_+ is computed using $\tilde{g}$ for analyzing the convergence of our algorithm. In other words, we only need to analyze the model-based regime.

4.1. Modified backtracking line search

In the model-based regime, $\tilde{g}$ is used in place of the exact gradient. As a result, the standard backtracking line search (BLS) may no longer terminate, as illustrated in the following example.

Example 1 *Consider $f(x) = (x-2)^2$ with a compensated model gradient $\tilde{g}(x) = 2x$. The compensated model gradient satisfies $\tilde{g}(x) > 0$ when $x \in (0, 2)$. Also, f is monotonically decreasing when $x < 2$, i.e., $f(x + \Delta x) \geq f(x)$ for any $\Delta x < 0$ and $x < 2$. So $f(x - \eta\tilde{g}(x)) \geq f(x) > f(x) - \alpha\eta\|\tilde{g}(x)\|^2$ for any $\alpha > 0$ and $\eta > 0$. In other words, the Armijo condition (Nocedal and Wright, 2006, page 33), which is used for terminating the standard BLS, cannot be satisfied for any positive η, implying that the standard BLS (Boyd and Vandenberghe, 2004) never terminates.*

To ensure that the BLS subroutine eventually terminates, we set a lower bound $\eta_{\min}$ for the step size η. The algorithm quits the BLS subroutine and reports that line search fails whenever $\eta \leq \eta_{\min}$. The modified BLS algorithm is presented as Algorithm 2.

We denote the initial step size in the BLS as $\eta_{\max}$. Upon successful termination, the modified BLS always returns a bounded step size $\eta \in (\eta_{\min}, \eta_{\max})$. The modified BLS will only be used in the model-based regime, whereas the standard BLS will be used in the model-free regime. In the following sections, we will simply use the term BLS when the type of BLS is clear from the context.

Algorithm 2 BTLINESEARCH

Input: $x, f, \Delta, \alpha, \beta, \eta_{\min}, \eta_{\max}$
Output: η

1: $\eta \leftarrow \eta_{\max}$
2: **while** $f(x - \eta\Delta(x)) > f(x) - \alpha\eta\|\Delta(x)\|^2$ and $\eta > \eta_{\min}$ **do**
3: $\quad \eta \leftarrow \beta\eta$
4: **end while**

Another issue we need to deal with, which will be shown in the next example, is that the algorithm may get stuck in the model-based regime and never converge to the optimal solution.

Example 2 *Consider the case in Example 1. The update rule for this case is given by $x_{j+1} = x_j - \eta\tilde{g}(x_j) = (1-2\eta_j)x_j$ for $j = 1, \ldots, N$. When the update starts at $x_1 < 0$ with $\eta_{\max} < 1/2$, we have $x_j = (1-2\eta_{\max})^j x_1 < 0$ for all j. Fix an $\alpha \in (0, 1/2)$, and suppose $\eta_{\min} < 1 - \alpha - 2/x_1$. It can be verified that BLS always succeeds since $f(x_{j+1}) < f(x_j) - \alpha\eta_j\|\tilde{g}(x_j)\|$ for $\eta_j \in (\eta_{\min}, 1/2)$. Despite the success of line search, the algorithm never converges to the optimal solution $x^\star = 2$.*

4.2. The sufficient decrease condition

Example 2 shows that the algorithm may not "make sufficient progress" even when line search succeeds. In the following, we formally define what "sufficient progress" means and give a verifiable condition to test whether the algorithm makes sufficient progress when line search succeeds.

Definition 1 *A point x_+ is said to satisfy the* sufficient decrease condition *for f at x if $f(x_+) < f(x) - t\|\nabla f(x)\|^2$ for some $t > 0$.*

In the vanilla gradient descent, when BLS succeeds, the Armijo condition, $f(x - \eta\nabla f(x)) < f(x) - \alpha\eta\|\nabla f(x)\|^2$, is satisfied for some step size $\eta > 0$ and $\alpha \in (0, 1/2)$, implying the sufficient decrease condition. However, the following example shows that the success of BLS does not in general imply the sufficient decrease condition when $\tilde{g}$ is used.

Example 3 *Consider again the case in Example 1. Set $x = -2$, $\alpha = 1/2$, and $\eta = 1/2$. BLS succeeds since $f(x - \eta\tilde{g}(x)) = 4 < 12 = f(x) - \alpha\eta\|\tilde{g}(x)\|^2$. Nevertheless, the sufficient decrease condition is not satisfied since $f(x - \eta\tilde{g}(x)) \geq 0 = f(x) - \alpha\eta\|\nabla f(x)\|^2$.*

Suppose $\tilde{g}$ is compensated at $\tilde{x}$. By Assumption 2, when $\tilde{x}$ is close to x, $\nabla r(\tilde{x})$ will be close to $\nabla r(x)$, implying that $\tilde{g}(x) = \nabla\hat{f}(x) + \nabla r(\tilde{x})$ will be close to $\nabla f(x) = \nabla\hat{f}(x) + \nabla r(x)$. When $\|\nabla f(x) - \tilde{g}(x)\|$ is sufficiently small, the following proposition shows that the success of BLS at x using $\tilde{g}$ implies the sufficient decrease condition.

Proposition 2 *Suppose BLS succeeds at x when using the compensated model gradient $\tilde{g}$, and $\tilde{g}$ satisfies $\|\nabla f(x) - \tilde{g}(x)\| \leq (1 - \gamma)\|\tilde{g}(x)\|/\gamma$ for some $\gamma \in (0, 1)$. Then the updated point $x_+ = x - \eta\tilde{g}(x)$, where η is given by BLS, satisfies the sufficient decrease condition for f at x.*

Define $e(x) \triangleq \|\nabla f(x) - \tilde{g}(x)\|$. According to Proposition 2, BLS will ensure a sufficient decrease for updates using $\tilde{g}$ when $e(x)$ is small. Conceivably, one can determine whether the sufficient decrease condition is still met through monitoring the value of $e(x)$. However, it is desirable to avoid evaluating $e(x)$ directly since computing ∇f is expensive. Instead of evaluating $e(x)$ directly, we propose to construct an upper bounding sequence $\{\bar{e}_j\}$ satisfying $\bar{e}_j \geq e(x_j)$ for all $j = 1, \ldots, N$ and monitor $\bar{e}_j$ in place of $e(x_j)$. Specifically, we construct $\{\bar{e}_j\}$ by choosing

$$\bar{e}_1 = 0, \qquad \bar{e}_{j+1} = L_r\|x_{j+1} - x_j\| + \bar{e}_j. \tag{5}$$

The following theorem shows that $\{\bar{e}_j\}$ constructed in (5) can be used to guarantee the sufficient decrease condition when BLS succeeds.

Theorem 3 *Consider a sequence $\{x_j\}_{j=1}^N$ defined by $x_{j+1} = x_j - \eta_j\tilde{g}(x_j)$, where η_j is given by BLS. Suppose BLS using $\tilde{g}(x_j)$ succeeds at x_j for all j. Let $\{\bar{e}_j\}$ be defined by (5). If for a fixed $\gamma \in (0, 1)$, we have*

$$\bar{e}_j \leq \frac{1 - \gamma}{\gamma}\|\tilde{g}(x_j)\| \tag{6}$$

for any j, then x_{j+1} satisfies the sufficient decrease condition for f at x_j for all $j = 1, \ldots, N$.

We can augment Algorithm 1 by incorporating condition (6) and use $\tilde{g}$ to update x only when (6) is satisfied and BLS succeeds. The factor γ can be viewed as a hyperparameter of the algorithm; a smaller γ will make (6) easier to satisfy, thus making our algorithm stay for more iterations in the model-based regime. The augmented algorithm is presented as Algorithm 3. As long as γ is small enough, condition (6) will always be satisfied when the algorithm stays in the model-based regime at x_j unless $\tilde{g}(x_j) = 0$. When $\tilde{g}(x_j) = 0$, it implies that x_j is a stationary point as predicted by $\tilde{g}$, at which point it becomes necessary to quit the model-based regime to verify whether x_j is truly a stationary point that satisfies $\nabla f(x_j) = 0$.

Algorithm 3 Gradient Compensation Algorithm

Start from $x^{(1)} = \hat{x}^\star$, $\bar{e}_0 = 0$ and $\delta_1 = \nabla r(\hat{x}^\star)$.
for $i = 1 : \texttt{iters}$ **do**
  $x^{(i+1)} \leftarrow \text{MODELBASEDDESCENT}(x^{(i)}, \delta_i, \gamma, \nabla\hat{f}, \bar{e}_0)$
  $\delta_{i+1} \leftarrow \nabla r(x^{(i+1)}) = \nabla f(x^{(i+1)}) - \nabla\hat{f}(x^{(i+1)})$
  $\eta \leftarrow \text{BTLINESEARCH}(x, f, \nabla f(x^{(i+1)}), \alpha, \beta, 0, \eta_{\max})$ ▷ Standard BLS
  $x_+^{(i+1)} \leftarrow x^{(i+1)} - \eta\nabla f(x^{(i+1)})$, $\bar{e}_0 \leftarrow L_r\|x_+^{(i+1)} - x^{(i+1)}\|$, $x^{(i+1)} \leftarrow x_+^{(i+1)}$
end for

Algorithm MODELBASEDDESCENT

Input: x, δ, γ, $\nabla\hat{f}$, $\eta_{\min}$, α, β, $\bar{e}_0$
Output: x

$\tilde{g} \leftarrow \nabla\hat{f} + \delta$, $\bar{e} \leftarrow \bar{e}_0$, $x_+ \leftarrow x$
while $\bar{e} \le \frac{1-\gamma}{\gamma}\|\tilde{g}\|$ and $\text{BTLINESEARCH}(x, f, \tilde{g}, \alpha, \beta, \eta_{\min}, \eta_{\max}) > \eta_{\min}$ **do**
  $x \leftarrow x_+$, $\tilde{g} \leftarrow \nabla\hat{f} + \delta$
  $\eta \leftarrow \text{BTLINESEARCH}(x, f, \tilde{g}, \alpha, \beta, \eta_{\min}, \eta_{\max})$ ▷ Modified BLS
  $x_+ \leftarrow x - \eta\tilde{g}$, $\bar{e} \leftarrow L_r\|x_+ - x\| + \bar{e}$
end while

4.3. Progress in the model-based regime

Theorem 3 shows that under condition (6), the success of BLS implies a sufficient decrease in the objective function. An immediate consequence of sufficient decrease is that the suboptimality gap in the model-based regime decreases geometrically, as described in the following proposition.

Proposition 4 *Suppose f satisfies Assumption 1, condition (6) holds, and the modified BLS using $\tilde{g}(x_j)$ succeeds at x_j for all $j = 1, 2, \dots N$. Then $f(x_{N+1}) - f^\star \le (1 - 2\mu\alpha\gamma^2\eta_{\min})^N (f(x_1) - f^\star)$, where $\eta_{\min}$ is the lower bound on the step size, and N is the number of iterations in the model-based regime between two model-free iterations.*

The convergence ratio in the model-based regime, according to Proposition 4, is given by $1 - 2\mu\alpha\gamma^2\eta_{\min}$. In the meantime, when f is assumed to be L_f-smooth (Karimi et al., 2016), the convergence ratio in the model-free regime is $1 - \mu/L_f$. When f is given, and the parameters of the algorithm are fixed, the progress made by the algorithm in either regime is a constant.

For quantifying the progress made per unit computational cost, we characterize the computational cost by the number of function evaluations rather than the number of iterations to better reflect the actual running time of the algorithm. In the model-based regime, because the model gradient has a closed-form expression (Assumption 3), the number of function evaluations per iteration is only determined by the number of line search attempts, which is upper bounded by $m_{\max} = \lceil(\log(\eta_{\min}/\eta_{\max}))/\log\beta + 1\rceil$ and hence does not depend on the problem dimension n. In comparison, for each iteration in the model-free regime, the computational cost grows with n due to the need for computing ∇f. Therefore, as n grows, the progress made per function evaluation in the model-based regime will eventually dominate that in the model-free regime, implying that using $\tilde{g}$ will decrease the function of evaluations needed for achieving the same level of suboptimality.

In Theorem 6 that will soon follow, we will show that the algorithm can ensure sufficient decrease for at least a few model-based iterations when BLS succeeds. Before presenting the theorem, we need to introduce the following definition.

Definition 5 (Bounded update direction) *An update direction mapping Δ is called a* bounded update direction *if there exist $\kappa_{\min}$ and $\kappa_{\max}$ satisfying $\kappa_{\min} \in (0, 1)$ and $\kappa_{\max} > \kappa_{\min}$ such that for any x and any bounded step size $\eta \in (\eta_{\min}, \eta_{\max})$, it holds that*

$$\kappa_{\min}\|\Delta(x)\| \leq \|\Delta(x_+)\| \leq \kappa_{\max}\|\Delta(x)\|,$$

where $x_+ = x - \eta\Delta(x)$.

Intuitively, Algorithm 3 will stay in the model-based regime when $\tilde{g}(x)$ is close to $\nabla f(x)$. To lower bound the number of iterations in the model-based regime, we need to bound the change of $\tilde{g}$ at each iterate, which is captured by the notion of bounded update direction. Using this notion, we present in the following theorem a lower bound on the number of model-based iterations.

Theorem 6 *Let x_1 be the initial point when Algorithm 3 enters the model-based regime with $\tilde{g}$. Suppose $\tilde{g}$ is a bounded updated direction mapping with constants $\kappa_{\max}$ and $\kappa_{\min}$, and $\tilde{g}(x_1) \neq 0$. Then the success of BLS implies the sufficient decrease condition if N satisfies*

$$\log|\kappa_{\max}^N - 1| + N\log\frac{1}{\kappa_{\min}} \leq \log\frac{1-\gamma}{\gamma\eta_{\max}L_r} + \log|\kappa_{\max} - 1| \tag{7}$$

when $\kappa_{\max} \neq 1$ and

$$\log N + N\log\frac{1}{\kappa_{\min}} \leq \log\frac{1-\gamma}{\gamma\eta_{\max}L_r} \tag{8}$$

when $\kappa_{\max} = 1$.

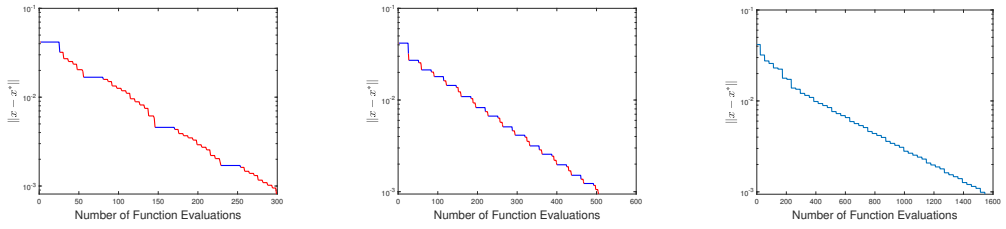
The left-hand side of (7) and that of (8) are monotonically increasing with respect to N. Let $N_{\min}$ be the largest N satisfying (7) or (8), depending on the value of $\kappa_{\max}$. Theorem 6 ensures that our algorithm makes sufficient progress in the model-based regime for at least $N_{\min}$ iterations as long as BLS succeeds. Because the right-hand side of (7) and (8) are monotonically decreasing as L_r grows, a smaller L_r is desired for making the algorithm stay for more iterations in the model-based regime when BLS succeeds. Recall in Section 2, we require h to be "small" without formally defining how to quantify the magnitude of h. Theorem 6 indicates that the L_r is a useful metric for quantifying the magnitude of h. Indeed, when the dynamics in (1) are nearly linear, the constant L_r will be small, and the algorithm is guaranteed to make better use of model information by staying in the model-based regime for more iterations. In the extreme case of $h \equiv 0$, the residual cost $r \equiv 0$ and hence $L_r = 0$, implying that the algorithm always stays in the model-based regime.

5. Numerical experiments

In this section, we shall refer to Algorithm 3 as GC, which stands for "gradient compensation." We compare GC with the method in Qu et al. (2020) in two different settings: 1) Both $\hat{f}$ and r are convex quadratic functions. 2) f is given by C in (3), and $\hat{f}$ is given by $\hat{C}$ from Section 2.

5.1. Quadratic functions

We consider quadratic functions $\hat{f}, r : \mathbb{R}^n \to \mathbb{R}$ defined as $\hat{f}(x) = c_1 x^\top P x/2$ and $r(x) = c_2(x - c_3)^\top Q(x - c_3)/2$, where both P and Q are positive definite and generated randomly. We chose $c_1 = 1$, $c_2 = c_3 = 0.1$, and $n = 4$. Thus, the smoothness factor L_r is given by $L_r = c_2 = 0.1$. The parameters of BLS were given by $\alpha = 0.3$, $\beta = 0.5$, $\eta_{\max} = 1$, and $\eta_{\min} = 0.005$.



(a) (b) (c)

Figure 1: Comparison among the convergence of (a) the GC algorithm with $\gamma = 0.6$, (b) the GC algorithm with $\gamma = 0.9$, and (c) the model-free method in Qu et al. (2020) for quadratic functions. For the GC algorithm, the iterations in the model-free regime are indicated in blue, and the ones in the model-based regime are indicated in red. When γ was chosen appropriately, the GC algorithm used fewer function evaluations than the model-free method because it used relatively inexpensive model-based iterations to make progress.

Results The results are shown in Fig. 1, where Fig. 1a and 1b are from the GC algorithm, and Fig. 1c is from the method by Qu et al. (2020). The blue part of the curve in Fig. 1a and 1b represents the model-free regime, and the red represents the model-based regime. The error $\|x - x^\star\|$ was only updated upon the completion of BLS. Since each test of the Armijo condition (Line 2 in Algorithm 2) in BLS requires one function evaluation, the error $\|x - x^\star\|$ sometimes remained unchanged over multiple consecutive function evaluations when BLS was in progress.

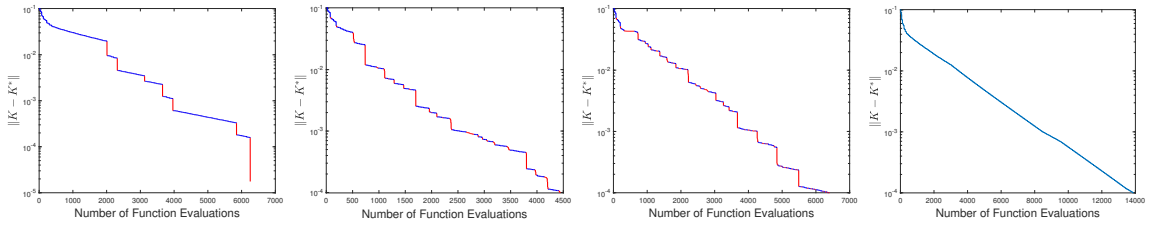
It can be seen that the GC algorithm required fewer function evaluations for achieving the same level of suboptimality. The choice of γ in (6) determines whether Algorithm 3 should continue to stay in the model-based regime. When a larger γ was used, the algorithm stayed for fewer iterations in the model-based regime because condition (6) became more difficult to satisfy.

5.2. Modified LQR problem

Consider the modified LQR problem defined by (1) and (2) in Section 2. The problem instance is similar to the one used in Qu et al. (2020), except that the dimension of the problem was set as $n = 4$ and $p = 3$, and the parameter ℓ in the nonlinear error term h was set as $\ell = 0.01$. The parameters of BLS were given by $\alpha = 0.3$, $\beta = 0.5$, $\eta_{\max} = 1$, and $\eta_{\min} = 0.05$. Unlike the quadratic functions studied in Section 5.1, in the setting of modified LQR, it is difficult to obtain the value of L_r. Instead, we treated L_r as a hyperparameter. The model gradient was computed analytically based on the result by Fazel et al. (2018). The optimal feedback gain $K^\star$ was computed approximately by running the model-free method in Qu et al. (2020) until $\|\nabla C(K^\star)\| < 10^{-4}$.

Effect of γ and L_r The choices of γ and L_r affect condition (6), which is used to monitor whether the sufficient decrease condition holds upon the success of BLS. When the minimum step size $\eta_{\min} = 0.05$ was used, both γ and L_r were found to have little effect on the performance of the GC algorithm. For instance, as γ varied between 0.0001 to 0.9999, the convergence of the GC algorithm

remained the same, as represented in Fig. 2a. Most of the time, the GC algorithm was found to leave the model-based regime almost immediately after entering. Moreover, the GC algorithm left the model-regime mostly due to the failure of BLS rather than violating condition (6). Since neither γ nor L_r affects BLS, this explains why changing γ and L_r did not help the GC algorithm stay in the model-based regime longer so as to reduce the number of function evaluations.



(a) (b) (c) (d)

Figure 2: Effect of $\eta_{\min}$ on the GC algorithm: (a) $\eta_{\min} = 0.05$. (b) $\eta_{\min} = 5 \times 10^{-4}$. (c) $\eta_{\min} = 5 \times 10^{-9}$. (d) $\eta_{\min} = 0.5$. (Blue: model-free regime. Red: model-based regime.) Choosing $\eta_{\min}$ either too small or too large increased the total number of function evaluations. When $\eta_{\min} = 0.5$, the GC algorithm only stayed in the model-free regime and behaved identically to the model-free method in Qu et al. (2020).

Effect of $\eta_{\min}$ Since BLS was what prevented the GC algorithm from staying in the model-based regime, we changed the minimum step size $\eta_{\min}$ from 0.05 to 5×10^{-4}. The result is shown in Fig. 2b, from which it can be seen that the algorithm stayed longer in the model-based regime, and the total number of function evaluations was smaller compared with Fig. 2a. However, upon further decreasing $\eta_{\min}$ to 5×10^{-9}, the total number of function evaluations increased, as shown in Fig. 2c. Recall that each attempt within BLS decreases the previous step size by a constant factor β and uses one function evaluation. For a smaller $\eta_{\min}$, in order to declare the failure of line search, BLS requires more attempts and hence more function evaluations before η reaches $\eta_{\min}$. To reduce the total number of iterations, one should choose $\eta_{\min}$ to balance between the number of iterations and the number of function evaluations (per iteration) in the model-based regime. Choosing a smaller $\eta_{\min}$ will make the algorithm stay in the model-based regime for more iterations and hence better exploit model information at the expense of more function evaluations per iteration.

We also experimented with a larger value of $\eta_{\min}$ by choosing $\eta_{\min} = 0.5$, the result of which is shown in Fig. 2d. As expected, a larger $\eta_{\min}$ caused the algorithm to spend fewer iterations in the model-based regime because the line search termination condition was tightened. In fact, for $\eta_{\min} = 0.5$, the GC algorithm behaved identically to the model-free method in Qu et al. (2020).

6. Conclusion

We have proposed a framework that combines model-based and model-free methods for optimal control. The framework formulates the optimal control problem as a composite optimization problem whose objective function is described by the sum of two terms: a model-based term obtained from an approximate model of the system dynamics and a residual term that captures the unknown modeling error. To solve the problem, we have developed a hybrid gradient-based algorithm that uses gradient from the approximate model with compensation from intermittent model-free updates. Both theoretical and numerical results show that our algorithm uses fewer function evaluations than a model-free policy gradient algorithm for reaching the same level of suboptimality.

References

Pieter Abbeel, Morgan Quigley, and Andrew Y. Ng. Using inaccurate models in reinforcement learning. In *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*, pages 1–8, Pittsburgh, Pennsylvania, 2006. ACM Press. ISBN 978-1-59593-383-6. doi: 10.1145/1143844.1143845.

C.G. Atkeson and J.C. Santamaria. A comparison of direct and model-based reinforcement learning. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 3557–3564 vol.4, April 1997. doi: 10.1109/ROBOT.1997.606886.

Amir Beck and Marc Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, January 2009. ISSN 1936-4954. doi: 10.1137/080716542.

Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004. ISBN 978-0-521-83378-3.

Jingjing Bu, Afshin Mesbahi, Maryam Fazel, and Mehran Mesbahi. LQR through the Lens of First Order Methods: Discrete-time Case. *arXiv:1907.08921 [cs, eess, math]*, July 2019.

Yevgen Chebotar, Karol Hausman, Marvin Zhang, Gaurav Sukhatme, Stefan Schaal, and Sergey Levine. Combining Model-Based and Model-Free Updates for Trajectory-Centric Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 703–711. PMLR, July 2017.

Xin Chen, Guannan Qu, Yujie Tang, Steven Low, and Na Li. Reinforcement Learning for Decision-Making and Control in Power Systems: Tutorial, Review, and Vision. *arXiv:2102.01168 [cs, eess]*, 2021.

Andrew R. Conn, Katya Scheinberg, and Luis N. Vicente. *Introduction to Derivative-Free Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, January 2009. ISBN 978-0-89871-668-9. doi: 10.1137/1.9780898718768.

Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. Gaussian Processes for Data-Efficient Learning in Robotics and Control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):408–423, February 2015. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2013.218.

Maryam Fazel, Rong Ge, Sham Kakade, and Mehran Mesbahi. Global Convergence of Policy Gradient Methods for the Linear Quadratic Regulator. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1467–1476. PMLR, July 2018.

Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear Convergence of Gradient and Proximal-Gradient Methods Under the Polyak-Łojasiewicz Condition. In Paolo Frasconi, Niels Landwehr, Giuseppe Manco, and Jilles Vreeken, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 9851, pages 795–811. Springer International Publishing, Cham, 2016. ISBN 978-3-319-46127-4 978-3-319-46128-1. doi: 10.1007/978-3-319-46128-1_50.

N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 3, pages 2619–2624 Vol.3, April 2004. doi: 10.1109/ROBOT.2004.1307456.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

Yansong Li and Shuo Han. Accelerating Model-Free Policy Optimization Using Model-Based Gradient: A Composite Optimization Perspective. *arXiv:2203.11424 [math]*, March 2022.

Yu. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013. ISSN 0025-5610, 1436-4646. doi: 10.1007/s10107-012-0629-5.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, New York, 2nd ed edition, 2006. ISBN 978-0-387-40065-5.

Neal Parikh and Stephen Boyd. Proximal Algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, January 2014. ISSN 2167-3888, 2167-3918. doi: 10.1561/2400000003.

Guannan Qu, Chenkai Yu, Steven Low, and Adam Wierman. Combining Model-Based and Model-Free Methods for Nonlinear Control: A Provably Convergent Policy Gradient Approach. *arXiv:2006.07476 [cs, eess, math]*, June 2020.

Benjamin Recht. A Tour of Reinforcement Learning: The View from Continuous Control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):253–279, 2019. doi: 10.1146/annurev-control-053018-023825.

R. Tyrrell Rockafellar. Monotone Operators and the Proximal Point Algorithm. *SIAM Journal on Control and Optimization*, 14(5):877–898, August 1976. ISSN 0363-0129. doi: 10.1137/0314056.

Shirli Di-Castro Shashua, Dotan Di Castro, and Shie Mannor. Sim and Real: Better Together. In *Thirty-Fifth Conference on Neural Information Processing Systems*, May 2021.

Stephen Tu and Benjamin Recht. The Gap Between Model-Based and Model-Free Methods on the Linear Quadratic Regulator: An Asymptotic Viewpoint. In *Proceedings of the Thirty-Second Conference on Learning Theory*, pages 3036–3083. PMLR, June 2019.

Cathy Wu, Aboudy Kreidieh, Kanaad Parvate, Eugene Vinitsky, and Alexandre M. Bayen. Flow: Architecture and Benchmarking for Reinforcement Learning in Traffic Control. *arXiv:1710.05465 [cs]*, 2017.