Thinking Fast and Slow: Optimization Decomposition Across Timescales

Gautam Goel Niangjun Chen Adam Wierman *

ABSTRACT

Many real-world control systems, such as the smart grid and software defined networks, have decentralized components that react quickly using local information and centralized components that react slowly using a more global view. This work seeks to provide a theoretical framework for how to design controllers that are decomposed across timescales in this way. The framework is analogous to how the network utility maximization framework uses optimization decomposition to distribute a global control problem across independent controllers, each of which solves a local problem; except our goal is to decompose a global problem temporally, extracting a timescale separation. Our results highlight that decomposition of a multi-timescale controller into a fast timescale, reactive controller and a slow timescale, predictive controller can be near-optimal in a strong sense. In particular, we exhibit such a design, named Multi-timescale Reflexive Predictive Control (MRPC), which maintains a per-timestep cost within a constant factor of the offline optimal in an adversarial setting.

1. INTRODUCTION

Modern control systems nearly always operate at multiple timescales. In the power grid, slow timescale economic dispatch is used to determine which baseload generators will supply power, while fast timescale frequency regulation is used to correct any imbalance between demand and supply that may arise [1]. In networking, software defined networks use a slow timescale "control plane" controller to decide where to send data packets, whereas fast timescale "data plane" controllers are responsible for routing the actual data [4].

Thus, the design and analysis of multi-timescale control systems has received considerable attention. However, the design of control policies for multi-timescale control systems typically does not address the joint problem of designing control policies across timescales. Instead, controllers for each timescale are designed independently. For example, in the power grid, the slow timescale problem of economic dispatch is usually studied separately from the fast timescale problem of frequency regulation. Recent work jointly designing economic dispatch and frequency regulation in the the power grid highlights significant inefficiency in designs that treated the two timescales independently [1].

In this work, our goal is to develop a framework for *deriving* rather than *assuming* a timescale separation in global optimization problems. In particular, we adapt the idea of optimization decomposition from the domain of distributed control into the domain of multi-timescale control.

There is a vast literature on optimization decomposition, in fields as diverse as Internet congestion control [5,8], smart grid control [1], and beyond [2]. The idea of this approach is to decompose a global optimization problem into smaller localized subproblems, each of which is solved by independent controllers. See [2] for a survey. In a similar way, our goal in this work is to look for decompositions of a global optimization problem in *time*, as opposed to in *space*.

However, this goal is made challenging by the tight coupling between the timescales due to the underlying dynamics of the system under consideration. Typically, spatial optimization decomposition is performed for static optimizations, but in multi-timescale control the dynamics of the system cannot be ignored. Any slow timescale action will impact the future state via the dynamics and hence must be taken into account when designing the fast controller; conversely, any fast timescale action impacts the state seen by a slow controller and thus impacts its design as well. This makes it unclear whether it is possible to achieve a clean separation between controllers at different timescales.

We make two main contributions in this work. We first introduce a simple but general model for studying multi-timescale optimal control. We consider a system subject to linear dynamics which is perturbed by noise; we make absolutely no assumptions about the nature of the noise, i.e., it may be random or even adversarial. This system can be controlled by two controllers, one of which is a traditional, "fast timescale" controller that can react immediately to the noise, and another which is a novel, "slow timescale" controller that is only able to react slowly, but which is empowered with access to more information than the fast controller and is potentially cheaper to use.

We then introduce a new multi-timescale control policy, MRPC, and derive strong guarantees on its performance. In particular, the per-step cost incurred by our algorithm is at most a constant more than that incurred by the offline optimal. The design of our policy is motivated by a

*This is an extended abstract of [3]. Readers can refer to the full version for more details and proofs.

Gautam Goel, Niangjun Chen, and Adam Wierman are with the Department of Computing and Mathematical Sciences, California Institute of Technology. Email:{ggoel, ncchen, adamw}@caltech.edu

structural result about the offline optimal control action, which highlights a strong decomposition between fast and slow timescale controllers. Applying this idea to the design of the online algorithm, we are able to achieve a clean separation between timescales. Remarkably, our decomposition results in a purely reflexive, "dumb" fast controller, which performs no optimization or lookahead. Thus, all of the computational burden is shifted onto the slow, "smart" controller. This property of MRPC is desirable in many applications since the slow controller is often centralized and able to take a global view of the system, but the fast controllers are decentralized and myopic, e.g., the power systems and networking examples mentioned above.

2. MODEL

Our goal is to study the design of controllers for systems that operate at multiple timescales. To this end, we focus on a simple but general optimal control problem.

The multi-timescale problem we consider builds on the following optimal control problem, which operates at a single timescale:

$$\begin{aligned} \min_{x,f} \quad & \sum_{t=1}^{T} c_x(x_t) + c_f(f_t) \qquad (1) \\ \text{s.t.} \quad & x_t = Ax_{t-1} + B^f f_t + w_t \\ & x_0 = 0 \end{aligned}$$

Here $x_t \in \mathbb{R}^n$ is the state variable, $f_t \in \mathbb{R}^n$ is the control action and $w_t \in \mathbb{R}^n$ is the disturbance. In our technical results, we assume that the control matrix B^f is invertible; considering the non-invertible case is an interesting direction for future work. The cost functions $c_x(\cdot), c_f(\cdot)$ are usually assumed to be non-negative and convex. The special case when each noise increment w_t is an i.i.d. Gaussian random variable and $c_x(\cdot), c_f(\cdot)$ are positive definite quadratic forms represents the Linear Quadratic Regulator (LQR) framework [7].

To extend (1) to a multi-timescale control problems, we introduce a "slow" controller. The slow controller reacts much less quickly to noise than the fast controller; however there are two potential benefits afforded by the existence of the slow controller.

First, in many situations the slow controller is centralized, and hence can use global information to make make better decisions than the decentralized, localized fast controllers. In our context, we model this by allowing the slow controller access to predictions of future noise increments. An example where the slow controller has this benefit is software defined networking, where the centralized controller has access to much more information than the local distributed controllers that provide congestion control via simple reactive policies [4]. Second, in many cases the slow controller is much cheaper to operate than the fast controller. Thus, making use of the slow controller is crucial for minimizing cost. For example, in the smart grid cheap "baseload" generators are used to supply the bulk of demand, whereas fast and relatively expensive "peaker" generators are used to quickly correct any imbalances between supply and demand that may arise [1, 6].

Adding a slow controller to the optimal control problem in (1) gives:

$$\begin{aligned} \min_{x,f,s} \quad & \sum_{t=1}^{T} c_x(x_t) + c_f(f_t) + c_s(s_t) \qquad (2) \\ \text{s.t.} \quad & x_t = Ax_{t-1} + B^f f_t + B^s s_t + w_t \\ & x_0 = 0 \\ & s_t = s_{t-1} \qquad \forall t \notin 0, k, 2k, \ldots \end{aligned}$$

Here s_t denotes the control action of a slow controller. The constraint on s_t means that the slow controller cannot react quickly, i.e., it can only change its action every k timesteps.

This formulation leads to a intrinsic notion of timescales: there is a *fast timescale* consisting of the timesteps $\{1, 2, \ldots\}$ in which the fast controller reacts, and a *slow timescale* consisting of the timesteps $\{1, k+1, 2k+1, \ldots\}$ at which the slow controller reacts. Clearly one could continue to add other timescales to this formulation as well, but we focus on the two timescale case for clarity.

The focus of this work is the design of a set of fast timescale and slow timescale controllers that operate independently but, together, approximate the optimal value of (2) without fully knowing the w_t's in advance. Motivated by the applications mentioned above, our goal is to develop designs where the slow controller is sophisticated and predictive, but the fast controller is simple and reactive.

Our results focus on settings where algorithms have access to a limited number of noisy predictions of future w_t. In particular, we assume that, at the start of each slow timescale interval, we have estimates $\hat{w}_t$ of the true noise increments over that slow timescale interval. Importantly, we do not make distributional assumptions about the predictions or prediction errors. In the full version of this work, we use a reduction from online convex optimization to argue that predictions are in fact necessary [3].

3. ARCHITECTURAL DECOMPOSITION FOR MULTI-TIMESCALE CONTROL

We now turn our attention to the joint multi-timescale control problem in (2), and focus on the co-design of fast and slow controllers. Recall that, while the slow controller cannot act as frequently, there are two benefits it usually provides: (i) it may have more information and computational power than the fast controller, e.g., in software defined networking and robotics, and (ii) it may be cheaper to operate than the fast controller, e.g., when scheduling generation in the smart grid. To capture these benefits of a slow controller, we consider a setting where the slow controller has access to noisy predictions but the fast controller does not. We also specifically highlight the case where the slow controller is cheaper to operate, though our results apply more generally.

Our main result in this section provides a performance bound for a new, near-optimal algorithm – *Multi-timescale Reflexive Predictive Control* (MRPC) – that consists of a simple, reflexive fast timescale controller and a predictive slow timescale controller. For concreteness and ease of presentation we focus on the case where the cost functions c_x, c_s, c_f in (2) are norms $\|\cdot\|_x, \|\cdot\|_s, \|\cdot\|_f$. Instead of merely presenting MRPC, we show that it can be derived naturally from understanding the structure of the offline optimal solution to (2).

The first step in the analysis is to establish a lower bound on the cost incurred by the offline optimal. As mentioned above, this lower bound highlights the decomposition between fast and slow used in the design of MRPC. We defer all proofs to the full version [3].

Lemma 1. *Letting OPT denote the optimal solution for* (2)*, we have:*

$$OPT \geq \min_s \sum_{r \in \mathcal{S}} \left[k\|s_r\|_s + C \sum_{t=r}^{r+k-1} \|(B^f)^{-1}(B^s s_t + w_t)\|_f \right]$$

where

$$C = \min\left(\frac{c}{2\|(B^f)^{-1}\|}, 1\right)$$

and c is a constant such that $\|v\|_x \geq c\|v\|_f$ *for all v.*

The lower bound has the following interpretation. Suppose the state is set at zero. After the slow controller has set its action to be s_r, the fast control action which corrects the remaining deviation from zero is $(B^f)^{-1}(B^s s_r + w_t)$, and our lower bound is the sum of the resulting costs (up to the constant C). Notice that the fast controller is extremely simple - all it does is continually correct any residual noise so that the state is always kept at zero. This is a crucial observation: *the form of the lower bound highlights a clear separation between a "smart", slow controller that does the planning and a "dumb" reactive fast controller.* This separation is then what we mimic in the design of MRPC. Informally, MRPC works as follows. Over each slow timescale slot, the slow controller greedily plays the slow control action which minimizes the expected cost using the predictions $\hat{w}_t$, under the assumption that the fast controller will keep the state at zero. As the true noise increments w_t are revealed one by one, the fast controller myopically corrects any noise so as to keep the state at zero.

Formally, let $\hat{f}$ and $\hat{s}$ denote the fast and slow control actions of MRPC. Then, the operation of each is as follows:

$$\hat{s}_r = \min_s \left[k\|s_r\|_s + \sum_{t=r}^{r+k-1} \|(B^f)^{-1}(B^s s_r + \hat{w}_t)\|_f \right] \quad (3)$$

$$\hat{f}_t = -(B^f)^{-1}(B^s \hat{s}_r + w_t) \qquad t = r, \ldots r+k-1 \quad (4)$$

Notice that the fast controller is very simple; it uses no predictions and performs no optimization. All of the prediction and optimization is shifted onto the slow controller. This is consistent with how the two controllers are used in many applications, where the slow controller is often centralized, with access to global information, but the fast controllers are usually decentralized, localized, and computationally limited. For example, in the smart grid a slow timescale global optimization problem is solved (economic dispatch) and then localized fast timescale controllers myopically correct any deviations that may arise (frequency regulation).

The following lemma provides an upper bound on the cost of MRPC.

Lemma 2.

$$MRPC \leq \min_s \sum_{r \in \mathcal{S}} \left[k\|s_r\|_s + \sum_{t=r}^{r+k-1} \|(B^f)^{-1}(B^s s_r + w_t)\|_f \right] + \|(B^f)^{-1}\| \sum_{t=1}^{T} \|\hat{w}_t - w_t\|_f$$

Together, our two lemmas immediately give a strong performance bound on MRPC:

Theorem 1. *Assume the cost functions* c_x, c_s, c_f *in* (2) *are norms* $\|\cdot\|_x, \|\cdot\|_s, \|\cdot\|_f$*. Then MRPC has an average per-stage cost within a constant factor of optimal. Specifically,*

$$\frac{MRPC}{T} \leq \max\left(\frac{2\|(B^f)^{-1}\|}{c}, 1\right)\frac{OPT}{T} + 2\|(B^f)^{-1}\|E(\hat{w}, w)$$

where c is a constant such that $\|v\|_x \geq c\|v\|_f$ *for all v and* $E(\hat{w}, w)$ *is the sample path average prediction error:*

$$E(\hat{w}, w) = \frac{1}{T}\sum_{t=1}^{T} \|\hat{w}_t - w_t\|_f$$

To get intuition for the bound itself, let us first look at the second term. The second term in the bound corresponds to the inefficiency due to noisy predictions. In particular, if we assume perfect lookahead (i.e $\hat{w}_t = w_t$ for all t), then the second term disappears. Thus, we see that prediction error has only an additive effect. It is important to realize that the analysis makes no modeling assumptions on the form of the prediction error. The error can be adversarial or stochastic and the result still holds.

The first term bounds the per-step cost incurred by our algorithm relative to the per-step cost incurred by the offline optimal. To get intuition for it, consider the case where control costs dominate the state costs. Specifically, consider the case where $c \geq 2\|(B^f)^{-1}\|$, and there are no errors in predictions. In this case, we have $MRPC = OPT$. It is worth highlighting this result in words: *when state costs dominate control costs and prediction errors are small, our distributed algorithm achieves the optimal value of* (2).

4. REFERENCES

[1] D. Cai, E. Mallada, and A. Wierman. Distributed optimization decomposition for joint economic dispatch and frequency regulation. In *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pages 15–22. IEEE, 2015.

[2] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, 2007.

[3] G. Goel, N. Chen, and A. Wierman. Thinking Fast and Slow: Optimization Decomposition Across Timescales. *arXiv:1704.07785*, Apr. 2017.

[4] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2015.

[5] S. H. Low, F. Paganini, and J. C. Doyle. Internet congestion control. *IEEE control systems*, 22(1):28–43, 2002.

[6] G. M. Masters. *Renewable and efficient electric power systems*. John Wiley & Sons, 2013.

[7] E. D. Sontag. *Mathematical control theory: deterministic finite dimensional systems*, volume 6. Springer Science & Business Media, 2013.

[8] R. Srikant. *The mathematics of Internet congestion control*. Springer Science & Business Media, 2012.