Direct Model Predictive Control

Shane Barratt
Department of Electrical Engineering
Stanford University
Stanford, CA 94305
sbarratt@stanford.edu

Abstract

Typically, models of dynamical systems are stated as recursive equations, that is, making multiple step predictions with them involves recursively applying themselves to their output. Such models are good for one-step ahead predictions, but when used for multi-step prediction, they suffer from compounding errors. When controlling dynamical systems, we seek a control sequence to apply the system, and therefore, we require dynamical system models that are accurate over multiple steps. In this paper, we suggest that one should use a direct forecasting model rather than a recursive forecasting model, and we interpret the resulting model predictive control optimization problem. The only downsides to using such a model are that it requires more storage and results in a dense (as opposed to sparse) optimization problem, both of which are not issues except for in extremely memory or compute-constrained applications.

1 Preliminaries

Many continuous systems are modeled using a discrete-time state-space model. In a fully-observed discrete-time state-space model, the state $x_t \in \mathbf{R}^n$ describes the system at time t and the control $u_t \in \mathbf{R}^m$ is applied to the system at time t. The system is assumed to evolve based on the following equation

$$x_{t+1} = f_t(x_t, u_t). \tag{1}$$

Assuming this model, optimal control laws can be found through model predictive control (MPC), which solves the following problem at each time step

$$\begin{array}{ll} \text{minimize} & c_f(x_N) + \sum_{\tau=0}^{N-1} c(x_\tau, u_\tau) \\ \text{subject to} & x_{t+1} = f_t(x_t, u_t),\ t = 0, \ldots, N-1, \\ & u_t \in \mathcal{U}_t,\ t = 0, \ldots, N-1, \\ & x_t \in \mathcal{X}_t,\ t = 0, \ldots, N, \\ & x_0 = x^{\text{init}} \end{array} \tag{2}$$

where $\{u_t\}_{t=0}^{N-1}$ are the optimization variables, c and c_f are stage costs, and the sets $\mathcal{U}_t$ and $\mathcal{X}_t$ represent constraints on the controls and states. If the dynamics are linear, *i.e.*, $f_t(x, u) = A_t x + B_t u$, the stage costs are convex, and the state and control constraints are convex, then the problem can efficiently be solved using convex optimization [1]. The resulting controller performs receding-horizon control; it observes x_t, solves the MPC problem, executes u_t on the system, observes x_{t+1}, solves the new MPC problem, and repeats. However, in real control problems, the system is neither linear nor time-invariant, and the system is stochastic, leading to model misspecification and hence a sub-optimal controller.

Submitted to ICML / IJCAI / AAMAS 2018 Workshop on Planning and Learning.

2 Problems with MPC

The main philosophy behind MPC is that we should choose controls to optimize the state trajectory as predicted by our model f_t. However, one of the main issues with MPC as stated in (2) is the way that it predicts future states. Its prediction for the state $\hat{x}_{t+i}$ is f recursively applied $i-1$ times. As a simple example, assume the true underlying model is $x_{t+1} = Ax_t$, *i.e.*, $B = 0$. Then if the learned dynamics are $\hat{A} = A + \Delta A$, the error in the prediction for time t starting at time 0 is

$$\begin{aligned} \|x_t - \hat{x}_t\|_2^2 &= \|A^t x_0 - (A + \Delta A)^t x_0\|_2^2 \\ &= \| \left[A^t - (A + \Delta A)^t\right] x_0\|_2^2. \end{aligned} \tag{3}$$

The A^t term cancels with the A^t term of $(A + \Delta A)^t$, but the remaining sum of matrices can lead to a large accumulation of errors, especially if A has singular values close to 1. This accumulation of errors is similar to what happens in numerical integration.

Suppose one has an observation of x_t and would like to select a matrix A to predict x_t from x_0. One could then define an objective

$$L(A) = \|x_t - A^t x_0\|_2^2,$$

but this objective is nonconvex in A, even for a single training example.

Further, not all linear mappings Cx_0 are representable by a matrix to an integer power. As a simple example, if $x_0 = 1$ and $x_2 = -5$, there is no real number c where $c^2 = -5$. This leads to significant bias in MPC's predictions for future states.

3 Direct Model Predictive Control

Instead of representing our prediction for future states iteratively, we propose that we should represent our prediction for future states as a direct model, *i.e.*, N separate models for each time horizon. At a state x_0, our prediction for the next N states are then

$$\begin{aligned} \hat{x}_1 &= f_1(x_0, u_0) \\ \hat{x}_2 &= f_2(x_0, u_0, u_1) \\ &\vdots \\ \hat{x}_N &= f_N(x_0, u_0, \ldots, u_{N-1}). \end{aligned} \tag{4}$$

This requires the construction of N *separate* models that predict all states in the prediction horizon as a function of the (current) state x_0 and control inputs up to that point in time. One way we can learn such a model after observing trajectories on the system is by setting up (and solving) N least-squares problems. We can integrate then these N prediction models into MPC via the following Direct MPC (DMPC) problem

$$\begin{aligned} \text{minimize} \quad & c_f(x_N) + \sum_{\tau=0}^{N-1} c(x_\tau, u_\tau) \\ \text{subject to} \quad & x_t = f_t(x_0, u_t, \ldots, u_{t+N-1}),\ t = 1, \ldots, N, \\ & u_t \in \mathcal{U}_t,\ t = 0, \ldots, N-1, \\ & x_t \in \mathcal{X}_t,\ t = 0, \ldots, N, \\ & x_0 = x^{\text{init}}. \end{aligned} \tag{5}$$

If l_f and l are convex, f_i are affine, and $\mathcal{U}_t$ and $\mathcal{X}_t$ are convex sets, we can efficiently solve for the global minimum (or verify infeasibility) of (5) using a convex solver, as was the case in MPC. If they are not, we can use a nonconvex solver, based on, *e.g.*, sequential convex programming, to find an (approximate) solution.

One important thing to note is that models representable by a recursive model are a subset of those representable by a direct prediction model. This is easy to see, by setting $f_1(x_0, u_0) = f(x_0, u_0)$, then setting $f_2(x_0, u_0, u_1) = f(f(x_0, u_0), u_1)$, and so on.

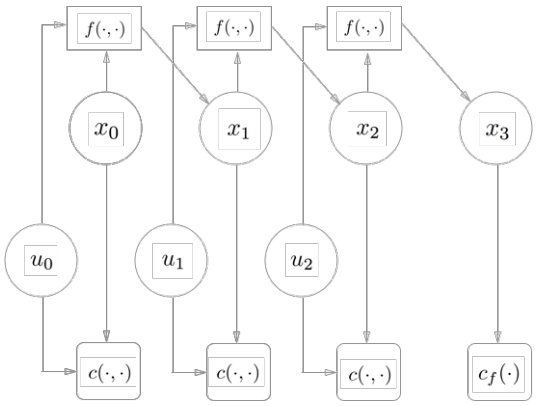

Figure 1: Variable dependency diagram for standard MPC problem (2).

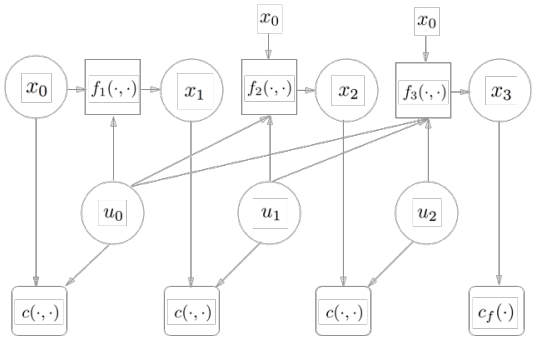

Figure 2: Variable dependency diagram for direct model predictive control problem (5).

Interpretations A great way to interpret the resulting optimization problem is visually. A variable dependency graph of the original MPC problem (2) for a time horizon of $N = 3$ is displayed in Figure 1. We focus on the dependency of the costs on the control variables u_0, u_1, u_2. The variable u_0 directly affects the first stage cost, then affects the next stage cost through f, then affects the rest of the costs by repeatedly going through the function f. This is contrasted with the variable dependency graph of (5) in Figure 2. In this case, u_0 directly affects the stage cost by being passed through f_1, f_2, and f_3, without any recursive applications of predictors. One direct advantage of this is that when computing derivatives of c with respect to one of the controls, we only need to differentiate through one prediction function, rather than up to N of them, which can lead to vanishing or exploding gradients and an unstable optimization procedure (particularly when the problem is nonconvex).

Training Difficulty? One potential counter-argument for the use of direct predictive models is that training N separate models will require N times the data. However, this is not the case. Suppose one has measured T pairs (x_t, u_t). Then to fit N models at different horizons, one sets up and solves N least-squares problems (all at different horizons). For example, the first problem is given by

$$\text{minimize} \sum_{t=1}^{T} \|x_{t+1} - f_1(x_t, u_t)\|_2^2$$

and the second is given by

$$\text{minimize} \sum_{t=1}^{T-1} \|x_{t+2} - f_2(x_t, u_t, u_{t+1})\|_2^2.$$

Each model f_i is trained using $T - i$ examples, so in fact, each model gets the same amount of data because it is reused.

4 Conclusion

In this paper, we described a modeling strategy for dynamical systems that circumvents the issue of compounding errors in recursive prediction models. We proposed a solution that relies on a *direct* prediction model, *i.e.*, it has a different model for each state in the horizon. Using a direct prediction model is strictly a generalization of a recursive prediction model. The only downsides are that it requires more storage and results in a dense (as opposed to sparse) optimization problem, both of which are not issues except for in extremely memory or compute-constrained applications. In future work, we hope to do an extensive comparison between MPC and DMPC across a variety of control tasks.

Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1656518.

References

[1] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, 2010.