INFINITE-HORIZON DIFFERENTIABLE MODEL PREDICTIVE CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper proposes a differentiable linear quadratic Model Predictive Control (MPC) framework for safe imitation learning. The infinite-horizon cost is enforced using a terminal cost function obtained from the discrete-time algebraic Riccati equation (DARE), so that the learned controller can be proven to be stabilizing in closed-loop. A central contribution is the derivation of the analytical derivative of the solution of the DARE, thereby allowing the use of differentiation-based learning methods. A further contribution is the structure of the MPC optimization problem: an augmented Lagrangian method ensures that the MPC optimization is feasible throughout training whilst enforcing hard constraints on state and input, and a pre-stabilizing controller ensures that the MPC solution and derivatives are accurate at each iteration. The learning capabilities of the framework are demonstrated in a set of numerical studies.

1 INTRODUCTION

Imitation Learning (IL, Osa et al., 2018) aims at reproducing an existing control policy by means of a function approximator and can be used, for instance, to hot-start reinforcement learning. Effective learning and generalisation to unseen data are paramount to IL success, especially in safety critical applications. Model Predictive Control (MPC, Maciejowski, 2000; Camacho & Bordons, 2007; Rawlings & Mayne, 2009; Kouvaritakis & Cannon, 2015; Gallieri, 2016; Borrelli et al., 2017; Raković & Levine, 2019) is the most successful advanced control methodology for systems with hard safety constraints. At each time step, a finite horizon forecast is made from a predictive model of the system and the optimal actions are computed, generally relying on convex constrained Quadratic Programming (QP, Boyd & Vandenberghe, 2004; Bemporad et al., 2000). Stability of the MPC in closed loop with the physical system requires the solution of a simpler unconstrained infinite horizon control problem (Mayne et al., 2000) which results in a value function (terminal cost and constraint) and a candidate terminal controller to be accounted for in the MPC forecasting. For Linear Time Invariant (LTI) models and quadratic costs, this means solving (offline) a Riccati equation (Kalman, 2001) or a linear matrix inequality (Boyd et al., 1994). Under these conditions, an MPC controller will effectively control a system, up to a certain accuracy, provided that uncertainties in the model dynamics are limited (Limon et al., 2009). Inaccuracies in the MPC predictions can reduce its effectiveness (and robustness) as the forecast diverges from the physical system trajectory over long horizons. This is particularly critical in applications with both short and long-term dynamics and it is generally addressed, for instance in robust MPC (Richards, 2004; Raković et al., 2012), by using a terminal controller to pre-stabilise the predictions.

This paper presents an infinite-horizon differentiable linear quadratic MPC that can be learned using gradient-based methods. In particular, the learning method uses an MPC controller where the terminal cost and terminal policy are the solution of an unconstrained infinite-horizon Linear Quadratic Regulator (LQR). A closed-form solution for the derivative of the Discrete-time Algebraic Riccati Equation (DARE) associated with the LQR is presented so that the stationary solution of the forward pass is fully differentiable. This method allows analytical results from control theory to be used to determine the stabilizing properties of the learned controller when implemented in closed-loop. The MPC QP is solved using the Operator Splitting Quadratic Program solver (OSQP, Stellato et al., 2017), a fast first-order QP solver that uses an implementation of the Alternating Direction Method of Multipliers (ADMM, Boyd, 2010). A method for differentiating the OSQP solution is also presented. Once the unconstrained LQR is computed, the predictive model is pre-stabilised using...
a linear state-feedback controller. This leaves the output of the MPC unaffected, but improves the conditioning of the solution. This in turn ensures that the gradients calculated in the backwards pass are accurate. Additionally, a further contribution is the MPC control formulation: a pre-stabilizing linear state-feedback controller is implemented from the solution of the DARE, and then the total control input is obtained as a perturbation of the feedback control law from the solution of a convex QP. The pre-stabilizing controller ensures that the QP is well conditioned and promotes a highly accurate global solution, which in turn ensures that the gradients calculated in the backwards pass are accurate. Additionally, an augmented Lagrangian penalty method is used to enforce constraints on state and control input. This method still enforces hard constraints as for a sufficiently large penalty term the constraints are strictly enforced, but ensures that the MPC problem is feasible throughout the training process. This is in contrast to (Amos et al., 2018), where a box differential dynamic programming method (Tassa et al., 2014) was implemented that could guarantee neither convergence nor feasibility.

The framework is implemented on a set of nominal systems, where it is demonstrated that the infinite horizon cost can be learned and the hard constraints can be guaranteed for a prediction horizon as short as six timesteps.

Notation $I_n$ is defined as the $n \times n$ identity matrix, $0_{m \times n}$ is defined as an $m \times n$ matrix of zeros, $0_n$ is defined as a vector of $n$ zeros, and $1_n$ is defined as a vector of $n$ ones. All inequalities $\leq$ and $\geq$ are considered element-wise in the context of vectors. $\rho(A)$ is defined as the spectral radius (largest absolute eigenvalue) of given matrix $A$. The operator $\mathtt{vec} : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{mn}$ is defined as $\mathtt{vec} \left( \begin{array}{c|c} c_1 & \cdots & c_n \end{array} \right) := (c_1 \cdots c_n)$, i.e. the columns of a matrix are stacked into a vector. For a matrix $A \in \mathbb{R}^{m \times n}$, the $V_{m,n} \in \mathbb{R}^{mn \times mn}$ permutation matrix is implicitly defined by $V_{m,n} \mathtt{vec} A := \mathtt{vec} A^\top$, and is unique. The Kronecker product is as defined in Magnus & Neudecker (1999) pp. 440.

2 Differentiable MPC

Linear quadratic MPC This paper considers linear time invariant systems of the form

$$x_{t+dt} = Ax_t + Bu_t,$$

where $x_t \in \mathbb{R}^n$ is the state system, $u_t \in \mathbb{R}^m$ is the control input, $A \in \mathbb{R}^{n \times n}$ is the state transition matrix, $B \in \mathbb{R}^{n \times m}$ is the input matrix, $t \in \mathbb{R}$ is the time, and $dt \in \mathbb{R}$ is the timestep (assumed constant). The control problem for this system is to determine the sequence of values of $u_t$ that achieve a desired level of performance (e.g. stability, frequency response, etc...), and when the system is subject to hard constraints on control input, $u_t \in \mathbb{U}$, and state, $x_t \in \mathbb{X}$, (or a combination of both), a well studied framework for controller synthesis is MPC. The principle of MPC is that the system’s control input and state are optimized over a finite prediction horizon, then the first element of the obtained control sequence is implemented at the current time step and the process is repeated ad infinitum. For linear MPC it is common to use a quadratic stage cost and box constraints on state and control ($x_L \leq x_k \leq x_U$ and $u_L \leq u_k \leq u_U$ where $u \leq 0 \leq u$), so that at each time index $t$ the vector of optimized control variables $u^*$ is obtained from
\[ \hat{u}_{0,N} = \arg\min_{\hat{u}} \frac{1}{2} \sum_{k=0}^{N-1} \hat{u}_k^\top R \hat{u}_k + \frac{1}{2} \sum_{k=1}^{N-1} \hat{x}_k^\top Q \hat{x}_k + \frac{1}{2} \hat{x}_N^\top Q_N \hat{x}_N + k_x \sum_{k=1}^{N} 1_m^\top r_k + k_u \sum_{k=0}^{N-1} 1_n^\top s_k \]

s.t. \( \hat{x}_0 = x_t, \)
\[
\hat{x}_{k+1} = A \hat{x}_k + B \hat{u}_k, \quad k \in \{0, \ldots, N-1\},
\]
\[
u - r_k \leq \hat{u}_k \leq \bar{\nu} + r_k, \quad k \in \{0, \ldots, N-1\},
\]
\[
r_k \geq 0, \quad k \in \{0, \ldots, N-1\},
\]
\[
\bar{\nu} - s_k \leq \hat{x}_k \leq \bar{\nu} + s_k, \quad k \in \{1, \ldots, N\},
\]
\[
 s_k \geq 0, \quad k \in \{1, \ldots, N\},
\]

where \( \hat{u}_{0,N} \) is the predicted control trajectory, \( \hat{x} \) is the predicted state trajectory, \( R \in \mathbb{R}^{m \times m} \geq 0 \) represents the stage cost on control input, \( Q \in \mathbb{R}^{n \times n} \geq 0 \) represents the stage cost on state, \( Q_N \in \mathbb{R}^{n \times n} \geq 0 \) represents the terminal cost on state, \( N \in \mathbb{N} \) is the prediction horizon, \( r_k \in \mathbb{R}^m \) are slack variables for the control constraint, \( s_k \in \mathbb{R}^n \) are slack variables for the state constraint, and \( k_x \in \mathbb{R} \) and \( k_u \in \mathbb{R} \) represent the cost of control and state constraint violations. The variables \( s \) and \( r \) enforce the box constraints on state and control using the augmented Lagrangian method (Nocedal & Wright, 2006, §17.2), and it can be shown that for sufficiently high \( k_x \) and \( k_u \) the constraints \( \underline{x} \leq \hat{x}_k \leq \bar{x} \) and \( \underline{s} \leq u_k \leq \bar{s} \) can be exactly guaranteed (Kerrigan & Maciejowski, 2000) (i.e. \( s = r = 0 \)). The benefit of this approach is that it ensures that the MPC optimization is feasible at each iteration of the learning process, whilst still ensuring that the constraints are ‘hard’.

To close the MPC control loop, at each timestep, \( t \), the first element of the optimized control sequence, \( \hat{u}_{0,t}^* \), is implemented as \( u_t \).

**Pre-stabilised MPC** If the control input is decomposed into \( u_t = K x_t + \delta u_t \), where \( K \in \mathbb{R}^{m \times n} \) is the linear state-feedback matrix and \( dt \) is a perturbation to the feedback control, system (1) becomes
\[
x_{t+dt} = (A + BK)x_t + B \delta u_t,
\]
and problem (2) becomes
\[
\delta u_{0,N}^* = \arg\min_{\delta} \frac{1}{2} \sum_{k=0}^{N-1} (K \hat{x}_k + \delta \hat{u}_k)^\top R (K \hat{x}_k + \delta \hat{u}_k) + \frac{1}{2} \sum_{k=1}^{N-1} \hat{x}_k^\top Q \hat{x}_k + \frac{1}{2} \hat{x}_N^\top Q_N \hat{x}_N
\]
\[
+ k_x \sum_{k=1}^{N} 1_m^\top r_k + k_u \sum_{k=0}^{N-1} 1_n^\top s_k
\]

s.t. \( \hat{x}_0 = x_t, \)
\[
\hat{x}_{k+1} = (A + BK) \hat{x}_k + B \delta \hat{u}_k, \quad k \in \{0, \ldots, N-1\},
\]
\[
u - r_k \leq \hat{u}_k \leq \bar{\nu} + r_k, \quad k \in \{0, \ldots, N-1\},
\]
\[
r_k \geq 0, \quad k \in \{0, \ldots, N-1\},
\]
\[
\bar{\nu} - s_k \leq \hat{x}_k \leq \bar{\nu} + s_k, \quad k \in \{1, \ldots, N\},
\]
\[
 s_k \geq 0, \quad \{1, \ldots, N\},
\]
so that \( \hat{u}_{0,t}^* = K x_t + \delta \hat{u}_0^* \) is implemented as \( u_t \). Using this decomposition, system (3) controlled with the solution of (4) is precisely equal to system (1) controlled with the solution of (2), but problem (4) is preferable from a computational standpoint if \( A \) is open-loop unstable (i.e. \( \rho(A) > 1 \)) and \( N \) is ‘large’, as this can lead to poor conditioning of the matrices defined in Appendix A. This is important in the context of differentiable MPC, as if \( A \) is being learned then there may be no bounds on its eigenvalues at any given iteration.

**MPC derivative.** Problems (2) and (4) can be rearranged into the QP form (see Appendix A for details)
\[
z^* = \arg\min_z \frac{1}{2} z^\top H z + q^\top z
\]
\[
s.t. l_b \leq M z \leq u_b.
\]
When $z^*$ is uniquely defined by (5), it can also be considered as the solution of an implicit function defined by the Karush-Kuhn-Tucker (KKT) conditions, and in Amos & Kolter (2017) it was demonstrated that it is possible to differentiate through this function to obtain the derivatives of $z^*$ with respect to the parameters $H, q, l, M, u$. The MPC controller can then be used as a layer in a neural network, and backpropagation can be used to determine the derivatives of an imitation cost function with respect to the MPC parameters $Q, R, A, B, u, \pi, x, \pi, k_x$ and $k_u$.

**Imitation Learning.** A possible use case of the derivative of a model predictive controller is imitation learning, where a subset of $\{\text{cost function, system dynamics, constraints}\}$ are learned from observations of a system being controlled by an ‘expert’. Imitation learning can be performed by minimizing the loss

$$
\frac{1}{T} \sum_{t=0}^{T} \| u_{t+t+Nd_t} - \hat{u}_{0,N}^*(x_t) \|^2_2 + \beta \| \hat{w}_t \|^2_2, \tag{6}
$$

where $u_t$ is the measured control input, $\hat{u}_{0,N}^*(x_t)$ is the full MPC solution, and $\beta \geq 0$ is a hyperparameter. It is assumed that both the learning algorithm and MPC controller have completely precise measurements of both the state and control input. The first term of (6) is the control imitation loss, and the second term penalises the one-step ahead prediction error

$$
\hat{w}_t = Ax_t + Bu_t - x_{t+d_t}.
$$

In practice, the prediction error loss might not be needed for the MPC to be learned correctly. Its use can however be instrumental for stability, as discussed in the next section.

## 3 TERMINAL COST FOR INFINITE HORIZON

**Terminal cost.** The infinite-horizon discrete-time Linear Quadratic Regulator (LQR) [Kalman 2001] is given with state feedback gain

$$
K = -(R + B^\top PB)^{-1}B^\top PA, \tag{7}
$$

where $P$ is obtained as a solution of the DARE

$$
P = A^\top PA - A^\top PB(R + B^\top PB)^{-1}B^\top PA + Q. \tag{8}
$$

The principle of the approach presented in this paper is the MPC controller (4) is implemented with $Q_N = P$. Proposition 1 summarises the relevant properties of the proposed MPC, building on classic MPC results from Scokaert & Rawlings (1998); Limon et al. (2003; 2009).

**Proposition 1.** Consider the MPC problem (4) with $Q_N = P$, where $P$ and $K$ solve (7). Define $V_N^*(x)$ as the optimal objective in (4) with $x_N = x$. Denote the optimal stage cost with $x_N = x$ as $\ell(x, \hat{u}_N^*(x)) = x^\top Q x + \hat{u}_N^*(x)^\top R \hat{u}_N^*(x)$. Then, for the closed-loop system, it follows that:

1. For any $N \geq 1$, there exists a closed and bounded set, $\Omega_N$, such that, if $x_0 \in \Omega_N$, then the MPC solution is infinite-horizon optimal for any $N \geq N$.
2. There exist positive scalars $d, \alpha$, such that, for any $N \geq 1$, the origin is asymptotically stable $\forall x_0 \in \Gamma_N$, with

$$
\Gamma_N = \{ x \in \mathbb{R}^n : V_N^*(x) \leq \ell(x, \hat{u}_N^*(x)) + (N - 1)d + \alpha \} . \tag{9}
$$

3. There exist a scalar, $\mu \geq 0$, such that, for any $N \geq 1$ the MPC is robustly feasible and the system is Input-to-State Stable (ISS) $\forall x_0 \in \Gamma_N$ given an additive model error, $\hat{w}$, such that: $\| \hat{w}_t \| \leq \mu, \forall t \geq 0$. In other words:

$$
V_N^*(x_{t+d_t}) \leq V_N^*(x_t) - \ell(x_t, \hat{u}_N^*(x_t)) + \sigma(\| \hat{w}_t \|),
$$

for some strictly increasing, bounded function, $\sigma(\cdot)$, with $\sigma(0) = 0$.

4. The QP matrices, $H, M$ and the vector $q$, in (5), have finite norms for any $N \geq 1$.

**Proof.** Proof of Proposition 1 is given in Appendix A.

1 Note that (5) differs from the form presented in Amos & Kolter (2017), and is instead the form of problem solved by the OSQP solver used in this paper. Appendix B demonstrates how to differentiate (5) using the solution returned by OSQP.
Implications. Proposition 1 has some important implications. First, point 1 implies that there exists a state-dependent finite horizon length, \( \bar{N} \), which is sufficient to make the MPC problem finite-horizon optimal. This \( \bar{N} \) can be upper bounded for a closed and bounded set of feasible states, \( \Omega_{\bar{N}} \). Scokaert & Rawlings (1998) proposed an iterative search that increases the horizon until optimality is verified; a similar algorithm is discussed in Appendix D where learning is completed iteratively reduced afterwards, although it is not implemented in this paper. Point 2, 3 state that MPC that can provide stability and constraints satisfaction, hence safety, if the model error is small. This also applies to small errors in the QP solution. Finally, point 4 states that the QP matrices have finite norm when the system dynamics are pre-stabilised using the LQR gain \( \bar{P} \) so the MPC problem is well conditioned and can be solved reliably to high accuracy. If the open-loop system is unstable then the terms of the matrices in Appendix A for the standard form are unbounded, so the QP solution may be poorly conditioned and the result inaccurate for long horizons. This can in turn invalidate the results of Amos & Kolter (2017) which assumes that the KKT conditions are exactly satisfied in order to compute its gradients. This method, on the other hand, improves the conditioning of the QP matrices and encourages a highly accurate solution, even over long horizons.

DARE Derivative. In order to implement \( Q_N = P \) in a differentiable imitation learning framework such as that presented in Section 2, the solution of the DARE must be differentiable w.r.t. its input matrices.

Proposition 2. Let \( P \) be the stabilizing solution of (8), and assume that \( Z_1^{-1} \) and \( (R + B^T PB)^{-1} \) exist, then the Jacobians of the implicit function defined by (8) are given by

\[
\frac{\partial \text{vec} P}{\partial \text{vec} A} = Z_1^{-1} Z_2, \quad \frac{\partial \text{vec} P}{\partial \text{vec} B} = Z_1^{-1} Z_3, \quad \frac{\partial \text{vec} P}{\partial \text{vec} Q} = Z_1^{-1} Z_4, \quad \frac{\partial \text{vec} P}{\partial \text{vec} R} = Z_1^{-1} Z_5,
\]

where \( Z_1, \ldots, Z_5 \) are defined by

\[
Z_1 := I_n^2 - (A^T \otimes A^T)[I_n^2 - (PB M_2 B^T \otimes I_n) - (I_n \otimes P B M_2 B^T) + (PB \otimes PB)(M_2 \otimes M_2)(B^T \otimes B^T)]
\]

\[
Z_2 := (V_{n,n} + I_n^2)(I_n \otimes A^T M_1)
\]

\[
Z_3 := (A^T \otimes A^T)[(PB \otimes PB)(M_2 \otimes M_2)(I_m^2 + V_{m,m})(I_m \otimes B^T P) - (I_n^2 + V_{n,n})(P B M_2 \otimes P)]
\]

\[
Z_4 := I_n^2
\]

\[
Z_5 := (A^T \otimes A^T)(PB \otimes PB)(M_2 \otimes M_2),
\]

and \( M_1, M_2, M_3 \) are defined by

\[
M_1 := P - P B M_2 B^T P, \quad M_2 := M_3^{-1}, \quad M_3 := R + B^T PB.
\]

Proof. If a stabilizing solution \( \rho(A + BK) \leq 1 \) to (8) exists, it is unique (Ionescu & Weiss 1992, Proposition 1), and the DARE can therefore be considered an implicit function of \( A, B, Q, \) and \( R \). Using the assumption that \( (R + B^T PB)^{-1} \) exists, it can be concluded that \( Z_1, \ldots, Z_5 \) and \( M_1, M_2, M_3 \) exist (the Kronecker product and matrix addition, subtraction, and multiplication always exist). Equation (8) can be given by

\[
P = A^T M_1 A + Q,
\]

which is differentiable, and \( M_1, M_2, M_3 \) are also differentiable. Differentials are taken for (10) and each of \( M_1, M_2, M_3 \) as

\[
d\text{vec} P = (V_{n,n} + I_n^2)(I_n \otimes A^T M_1)d\text{vec} A + (A^T \otimes A^T)d\text{vec} M_1 + d\text{vec} Q
\]

\[
d\text{vec} M_1 = [I_n^2 - (PB M_2 B^T \otimes I_n) - (I_n \otimes P B M_2 B^T)]d\text{vec} P
\]

\[
- (PB \otimes PB)d\text{vec} M_2 - (I_n^2 + V_{n,n})(P B M_2 \otimes P)d\text{vec} B
\]

\[
d\text{vec} M_2 = -(M_2 \otimes M_2)d\text{vec} M_1
\]

\[
d\text{vec} M_3 = d\text{vec} R + (B^T \otimes B^T)d\text{vec} P + (I_m^2 + V_{m,m})(I_m \otimes B^T P)d\text{vec} B,
\]

Note that any stabilising gain would be acceptable for the purpose of QP conditioning only.
then these can be combined using the differential chain rule \cite{Magnus&Neudecker1999} Theorem 18.2) to obtain
\[ Z_1 \text{dvec}P = Z_2 \text{dvec}A + Z_3 \text{dvec}B + Z_4 \text{dvec}Q + Z_5 \text{dvec}R. \]

The Jacobians, as defined in Proposition 2, therefore exist if \( Z_1^{-1} \) exists. \(\square\)

The sensitivity of the solution of the DARE has been investigated in the context of robustness to perturbations in the input matrices, e.g. \cite{Sun1998, Konstantinovetal1993}, and the analytical derivative of the continuous-time algebraic Riccati equation was derived in \cite{Brewer1977} by differentiating the exponential of the Hamiltonian matrix, but to the best of the authors’ knowledge this is the first presentation of an analytic derivative of the DARE using the differential calculus approach of \cite{Magnus&Neudecker1999}.

Algorithm overview Algorithm 1 presents the overall procedure for learning a subset, \( S \), of the MPC controller parameters, \( \mathcal{M} = \{ A, B, Q, R, x, \pi, k_u, k_x \} \), with the key steps of the forwards and backwards pass of a gradient based optimization method. It is important to remark that, in each forward pass the MPC terminal cost matrix, \( Q_N \), and the pre-stabilizing controller, \( K \), are set from the solution of the DARE. The DARE and the MPC QP solutions are then differentiated in the backward pass to provide the gradients used in the update step, and the update can be performed using any gradient-based method. It is also worth noting that the horizon, \( N \), is not differentiable, and that simultaneously learning the entire set \( \mathcal{M} \) is challenging in general.

Algorithm 1 Infinite-horizon MPC Learning

<table>
<thead>
<tr>
<th>In: ( \mathcal{M} \setminus S, N &gt; 0, \beta &gt; 0, N_{\text{epochs}} &gt; 0 )</th>
<th>Out: ( S ) for ( i = 0...N_{\text{epochs}}, \text{do} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward Pass</td>
<td>( (K, P) \leftarrow \text{DARE} ) \cite{Brewer1977} solution ( Q_T \leftarrow P ) ( \hat{u}_{0,N} \leftarrow \text{MPC QP} ) \cite{Magnus&amp;Neudecker1999} solution ( L \leftarrow \text{Imitation loss} ) \cite{Konstantinovetal1993}</td>
</tr>
<tr>
<td>Backward Pass</td>
<td>Differentiate loss \cite{Sun1998} Differentiate MPC QP solution, ( \hat{u}_{0,N} ), using Appendix \cite{Magnus&amp;Neudecker1999} Differentiate DARE, ( (P, K) ), using Proposition 2</td>
</tr>
<tr>
<td>Update step</td>
<td>( S \leftarrow \text{Gradient-based step} )</td>
</tr>
</tbody>
</table>

4 NUMERICAL EXPERIMENTS

Data generation Data for an expert controller was generated with a second-order mass-spring-damper model parameterized by a mass, \( m \in \mathbb{R} > 0 \), damping coefficient, \( c \in \mathbb{R} \), stiffness, \( k \in \mathbb{R} \), and timestep \( dt \in \mathbb{R} > 0 \), where
\[
A = \exp(A_c dt), \quad A_c = \begin{bmatrix} 0 & \frac{1}{m} \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix}, \quad B = (A - I_n)A_c^{-1}B_c, \quad B_c = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}.
\]

For this system \( x_t \in \mathbb{R}^2 \) represents the position and velocity of the mass, and the control input \( u_t \in \mathbb{R} \) represents a force applied to the mass. Seven models were created with the mass, stiffness, and timestep fixed at \( m = 1, k = 1, \) and \( dt = 0.2 \), and \( c \) was varied as shown in Table 1 to affect the open-loop stability of the models (\( c > 0 \) implies stable, \( c < 0 \) implies unstable). The expert data was then generated by simulating each of the systems the initial condition \( x_0 = (0, 3) \) in closed-loop with an infinite-horizon MPC controller (i.e. the horizon was increased until the open-loop state predictions matched the closed-loop response), using \( Q = \text{diag}(1, 1) \), \( R = 2 \), \( (u, \pi) = (-\infty, 0.5) \), \( e = (-1, -\infty) \), and \( \pi = (1, \infty) \). The constraint set was chosen so that the constraints on both state and control input were strongly active at the solution whilst ensuring that the ‘expert’ MPC optimization was feasible. The values \( k_u = k_x = 100 \) were found to be sufficient to enforce the constraints on control and state, and were used for all experiments.

<table>
<thead>
<tr>
<th>System</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c )</td>
<td>1</td>
<td>0.5</td>
<td>0.1</td>
<td>-0.1</td>
<td>-0.3</td>
<td>-0.5</td>
<td>-0.6</td>
</tr>
</tbody>
</table>

Table 1: Damping coefficient \( c \) used to generate the seven imitation systems.
Learning The learner and expert shared all system and controller information apart from the state transition matrix $A$, which was learned, and the MPC horizon length, which was implemented as each of $N \in \{2, 3, 6\}$ in three separate experiments. $A$ was initialized with the correct state transition matrix plus a uniformly distributed pseudo-random perturbation in the interval $[-0.5, 0.5]$ added to each element. The learner was supplied with the first 50 elements of the closed loop state trajectory and corresponding controls as a batch of inputs, and was trained to minimize the imitation loss (6) with $\beta = 0$, i.e. the state dynamics were learned using predicted control trajectories only, and the state transitions are not made available to the learner (this is the same approach used in [Amos et al., 2018]. The experiments were implemented in Pytorch 1.2.0 using the built-in Adam optimizer [Kingma & Ba, 2014] for 1000 steps using default parameters. The MPC optimization problems were solved for the ‘expert’ and ‘learner’ using OSQP with settings (eps_abs=1E-10, eps_rel=1E-10, eps_rim_inf=1E-10, eps_dual_inf=1E-10).

Training Results Figure 1 shows the imitation and model loss at each of the 1000 optimization iterations for each of the tested horizon lengths. It can be seen that all of the generated systems are ‘trainable’ for all MPC horizon lengths, in the sense that the imitation loss converges to a low value, although the imitation loss converges to a local minimum in general. In most cases, the learned model converges to a close approximation of the real model, although as the problem is non-convex this cannot be guaranteed, and it is also shown that there are some cases in which the model does not converge correctly. This occurred exclusively for $N = 2$, where neither system 4 nor system 2 converge to the correct dynamics. Additionally, it can be seen that both the imitation loss and model loss converge faster as the prediction horizon is increased. This suggests that a longer learning horizon improves the learning capabilities of the methods, but there is not sufficient data to demonstrate this relationship conclusively.

Testing Results To test generalization performance, each of the systems was re-initialized with initial condition $x_0 = (0.5, 2)$ and simulated in closed loop using the learned controller for each
horizon length. The results are compared in Figure 2 against the same systems controlled with an infinite horizon MPC controller. The primary observation is that as the learned MPC horizon is increased to $N = 6$, the closed loop trajectories converge to expert trajectories, indicating that the infinite horizon cost has been learned (when using the infinite horizon cost with no model mismatch or disturbance, the predicted MPC trajectory is exactly the same as the closed loop trajectory), and that the state constraints are guaranteed for $N \geq 4$. Furthermore, it can be seen that the learned controllers are stabilizing, even for the shortest horizon and the most unstable open-loop systems. This is also the case for systems 2 and 4 where the incorrect dynamics were learned, although in this case the state constraints are not guaranteed for $N = 2$.

![Figure 2: Closed-loop trajectories using the expert and learned controllers. Trajectories only shown for $x_1$ (i.e. position), but $x_2$ (i.e. velocity) can be inferred. Expert controllers shown with solid lines, and learned controller shown with dotted lines. The hard constraints on state are shown in the red regions.](image.png)

**Limitations** The major theoretical limitation of the above approach is the restriction to LTI systems. A more comprehensive solution would cover linear time varying systems (for which the MPC is still obtained from the solution of a QP), however in this case the infinite horizon cost cannot be obtained from the solution of the DARE, and the extension of the methods presented in this paper to time varying or non-linear models is non-trivial (see Appendix E for further discussion). There are also implementation issues with the proposed algorithm. The derivative of the DARE presented in Proposition 2 involves multiple Kronecker products and matrix inversions (including an $n^4 \times n^4$ matrix inversion) that do not scale well to large state vectors, although the dynamics of physical systems can usually be reasonably approximated with only a handful of state variables, so this may not become an issue in practice. The algorithm also relies on the existence of a stabilizing solution to the DARE. Theories for the existence of stabilizing solutions of the DARE are non-trivial (e.g. Ran & Vreugdenhil [1988]), and it is not immediately obvious how to enforce their existence throughout the training process (stabilizibility can be encouraged using the one-step ahead term in 6).

5 CONCLUSION

This work presented a method to differentiate through an infinite-horizon linear quadratic MPC, where the solution of the DARE was used to compute a terminal cost from the MPC optimization problem. The final control sequence is obtained from the solution of a QP that is structured so that its always both well-conditioned and feasible, and the whole forward pass is end-to-end differentiable, so can be included as a layer in a neural network architecture. The approach was demonstrated on a set of imitation learning experiments for a family of ‘expert’ controlled second-order systems with different stability properties. In particular, it is shown that a short prediction horizon can be found such that the resulting MPC is stable and infinite-horizon optimal.
REFERENCES


Savecsa V. Raković and William S. Levine (eds.). *Handbook of Model Predictive Control*. Springer International Publishing, 2019. doi: 10.1007/978-3-319-77489-3. URL [https://doi.org/10.1007/978-3-319-77489-3](https://doi.org/10.1007/978-3-319-77489-3)


APPENDICES

A MPC QUADRATIC PROGRAM

Problem (2) is equivalent to

\[
\begin{align*}
    z^* &= \arg \min_z \frac{1}{2} z^\top \begin{bmatrix} R + \Psi^\top Q \Psi & 0_{N_m \times N_m} \\ 0_{N_m \times N_m} & 0_{N_n \times N_n} \end{bmatrix} z + \begin{bmatrix} \Psi Q \Phi x_t \end{bmatrix}^\top z \\
    \text{s.t.} \quad & \begin{bmatrix} u \\ -\infty \\ 0_{N_m} \\ \tilde{x} - \Phi x_t \\ -\infty \\ 0_{N_n} \end{bmatrix} \leq \begin{bmatrix} I_{N_m} \\ I_{N_m} \\ -I_{N_m} \\ I_n \\ -I_n \end{bmatrix} z \leq \begin{bmatrix} \infty \\ \infty \\ \infty \\ \infty \\ \infty \end{bmatrix} \\
\end{align*}
\]

where

\[
\Psi = \begin{bmatrix} B \\ \vdots \\ A^{N-1}B \end{bmatrix}, \quad \Phi = \begin{bmatrix} \Psi \\ \vdots \\ A \end{bmatrix}, \quad \Psi = \begin{bmatrix} 0_{n \times N_n} \\ \vdots \\ B \end{bmatrix}
\]

are of conformal dimensions. Using the above, problem (4) is then equivalent to

\[
\begin{align*}
    z^* &= \arg \min_z \frac{1}{2} z^\top \begin{bmatrix} (K \hat{\Psi} + I_{N_m})^\top R (K \hat{\Psi} + I_{N_m}) + \hat{\Psi}^\top \hat{Q} \hat{\Psi} \\ 0_{N_m \times N_m} \end{bmatrix} z \\
    &+ \begin{bmatrix} (K^\top R (K \hat{\Psi} + I_{N_m}) + \hat{Q} \hat{\Psi})^\top \hat{\Phi} x_t \end{bmatrix}^\top z \\
    \text{s.t.} \quad & \begin{bmatrix} u - K \hat{\Phi} x_t \\ -\infty \\ 0_{N_m} \\ \tilde{x} - \Phi x_t \\ -\infty \\ 0_{N_n} \end{bmatrix} \leq \begin{bmatrix} (K \hat{\Psi} + I_{N_m}) \\ (K \hat{\Psi} + I_{N_m}) \\ \Psi \\ \Psi \\ I_{N_n} \\ -I_n \end{bmatrix} z \leq \begin{bmatrix} \infty \\ \infty \\ \infty \\ \infty \\ \infty \end{bmatrix} \\
\end{align*}
\]

where now

\[
\begin{align*}
    z &= \begin{bmatrix} \delta u \\ r \\ s \end{bmatrix}, \quad \Phi = \begin{bmatrix} (A + BK) \\ \vdots \\ (A + BK)^{N-1}B \end{bmatrix}, \quad \Psi = \begin{bmatrix} B \\ \vdots \\ (A + BK)^N \end{bmatrix}
\end{align*}
\]

and

\[
\begin{align*}
    Q &= \begin{bmatrix} 0_{n \times n} \\ Q \end{bmatrix}, \quad \hat{\Phi} = \begin{bmatrix} I_n \\ \Phi \end{bmatrix}, \quad \hat{\Psi} = \begin{bmatrix} I_{N_n} \\ \Psi \end{bmatrix}, \quad K = \begin{bmatrix} K \\ \vdots \\ K \end{bmatrix}
\end{align*}
\]

are of conformal dimensions.
B OSQP DERIVATIVES

OSQP solves quadratic programs of the form (5), and returns values for $z$, $y$, and $s$ that satisfy

$$
Mz = s, \\
Hz + q - M^T y = 0, \\
s \in \mathcal{C}, \quad y \in \mathcal{N}_\mathcal{C}(s),
$$

(Stellato et al., 2017, §2), where $\mathcal{C}$ is the set $\{s : 1 \leq s \leq u\}$, and $\mathcal{N}_\mathcal{C}$ is the normal cone of $\mathcal{C}$. The values of $y$ that are returned by the solver can be used to determine whether the constraints are strongly active at the solution, where $y_i = 0$ indicates that the constraints $l_i \leq M_i z$ and $M_i z \leq u_i$ are inactive, $y_i > 0$ indicates that $M_i z \leq u_i$ is strongly active, and $y_i < 0$ indicates that $l_i \leq M_i z$ is strongly active. The solution can therefore be completely characterised by the KKT system

$$
\begin{bmatrix}
H & M^T \\
M_L & M_C \\
\end{bmatrix}
\begin{bmatrix}
z \\
\bar{u}_L \\
\end{bmatrix}
= 
\begin{bmatrix}
q \\
\bar{u}_L \\
\end{bmatrix}
$$

where $\mathcal{U} = \{i : y_i > 0\}$ and $\mathcal{L} = \{i : y_i < 0\}$, and the notation $M_S$ indicates a matrix consisting of the $i \in \mathcal{S}$ columns of given matrix $M$, and $v_S$ indicates a vector consisting of the $i \in \mathcal{S}$ elements of given vector $v$. Equation (11) can then be differentiated using the techniques detailed in (Amos & Kolter, 2017, §3)

C PROOF OF PROPOSITION 1

Proof. (Proposition 1) The first point follows from (Scokaert & Rawlings, 1998). The next two points of Proposition 1 stem from the results in (Limon et al., 2003, 2009). In particular, the closed-loop is Lipschitz since the model is linear and the controller is the solution of a strictly convex QP. Moreover, the LQR provides a contractive terminal set. The final point follows from the fact that $(A + BK)^N$ has eigenvalues in the unit circle, $\forall N \geq 0$. Proof of point 4 is concluded by inspection of the QP matrices (Appendix A) and by application of Theorem 5.6.12, page 298 of Horn & Johnson (2012) which states that, a given a bound, $\bar{p}$, on the spectral radius, then there exists a matrix norm which is also less than $\bar{p}$. \qed

D VERIFICATION AND REDUCTION OF THE PREDICTION HORIZON

A method is proposed for the reduction of the MPC prediction horizon after imitation learning. The idea is to be able to reproduce the infinite-horizon optimal MPC up to a tolerance $\epsilon$ with high probability. Do so, we check that, for a candidate horizon $\bar{N}$, the MPC action deltas, $\delta u_k^\star$, satisfy $\|\delta u_k^\star\| \leq \epsilon$, for all $k \geq \bar{N}$. This means that the optimal action is equal to the LQR up to a tolerance $\epsilon$. In order to provide a high probability guarantee of this condition, we propose the use of a probabilistic verification approach, similar to Bobiti (2017). This is described in Algorithm 2. In particular, the condition is checked on a high number, $n_s$, of initial states. These states are sampled uniformly from a set of interest $\mathcal{X}$, which can be either the state constraints $\mathcal{X}$ or an estimate of the region of attraction, $\Gamma_N$. If verified, this set is a region of attraction for the system with high probability. The relationship between the number of samples and the verification probability is discussed in (Bobiti, 2017, Chapter 5). The algorithm also checks whether the infinite horizon condition has been reached for the $\bar{N}$ used during training. Finally, a line search for a suitable $\mathcal{X}$ is proposed using a scaling factor $\eta \in (0, 1)$. In particular, the

<table>
<thead>
<tr>
<th>Algorithm 2 MPC horizon verification and reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In:</strong> $\bar{N} &gt; 0$, $\mathcal{X}_0 \subseteq \mathcal{X}$, $\mathcal{M}$, $(P, K)$ from (7-8), $\epsilon &gt; 0$, $n_s &gt; 0$, $\eta \in (0, 1)$.</td>
</tr>
<tr>
<td><strong>Out:</strong> $\bar{N}$, $\mathcal{X}$</td>
</tr>
<tr>
<td>$\mathcal{X} \leftarrow \mathcal{X}_0$ while $\mathcal{X} \supseteq \emptyset$ do</td>
</tr>
<tr>
<td>$\bar{N} \leftarrow \bar{N}$ while $\bar{N} &gt; 0$ do</td>
</tr>
<tr>
<td>$\mathcal{X}_{\text{sample}} \leftarrow n_s$ uniform state samples, s.t. $x \in \mathcal{X}$</td>
</tr>
<tr>
<td>$\delta u^\star \leftarrow$ Solution of MPC QP (11), $\forall x \in \mathcal{X}<em>{\text{sample}}$ if $|\delta u^\star(x)| \leq \epsilon, \forall k \geq \bar{N}$, $\forall x \in \mathcal{X}</em>{\text{sample}}$ then</td>
</tr>
<tr>
<td>$\bar{N} \leftarrow \bar{N} - 1$</td>
</tr>
<tr>
<td>$\mathcal{X} \leftarrow \eta \mathcal{X}$</td>
</tr>
<tr>
<td>Procedure failed</td>
</tr>
<tr>
<td>$\bar{N} \leftarrow \bar{N} + 1$</td>
</tr>
<tr>
<td>Go to Algorithm 1</td>
</tr>
</tbody>
</table>

12
initial set is downscaled until either an horizon is found or the set becomes empty. In latter case the search fails and the procedure returns to the training algorithm with an increased $N$. Noticeably, the proposed algorithm does not require to explicitly compute the terminal set in which the LQR is invariant and it could be used also for non-linear MPC if an infinite-horizon (or a stabilising) terminal controller is available.

**E  Nonlinear models**

As discussed in the main paper, our approach is currently limited to Linear Time Invariant (LTI) systems. In general, conditions for infinite-horizon optimality of systems that are not LTI are non-trivial. Some of the results on MPC stability could however be maintained, for example in the case when the LQR value function, $x^TPx$, is a local control Lyapunov function (Khalil 2014; Mayne et al. 2000). In this case, the stability and intrinsic robustness results are maintained (see Limon et al. 2003; 2009). For these systems, it would be possible to use our method, for instance in combination with Amos et al. (2018), to provide a stable Non-linear MPC. This is however a big assumptions for systems that are very non-linear. Assessing this LQR controllability condition could be done, for instance, by training a local linear model around the target equilibrium (origin) and then checking whether the DARE is solvable. This should be performed before starting the imitation learning. We leave the study of more general systems to future work.