Robotics: Science and Systems 2020
Corvalis, Oregon, USA, July 12-16, 2020

Deep Learning Tubes for Tube MPC

David D. Fan[1,2], Ali-akbar Agha-mohammadi[2], and Evangelos A. Theodorou[1]

***Abstract*—Learning-based control aims to construct models of a system to use for planning or trajectory optimization, e.g. in model-based reinforcement learning. In order to obtain guarantees of safety in this context, uncertainty must be accurately quantified. This uncertainty may come from errors in learning (due to a lack of data, for example), or may be inherent to the system. Propagating uncertainty forward in learned dynamics models is a difficult problem. In this work we use deep learning to obtain expressive and flexible models of how distributions of trajectories behave, which we then use for nonlinear Model Predictive Control (MPC). We introduce a deep quantile regression framework for control that enforces probabilistic quantile bounds and quantifies epistemic uncertainty. Using our method we explore three different approaches for learning tubes that contain the possible trajectories of the system, and demonstrate how to use each of them in a Tube MPC scheme. We prove these schemes are recursively feasible and satisfy constraints with a desired margin of probability. We present experiments in simulation on a nonlinear quadrotor system, demonstrating the practical efficacy of these ideas.**

I. Introduction

In controls and planning, the idea of adapting to unknown systems and environments is appealing; however, guaranteeing safety and feasibility in the midst of this adaptation is of paramount concern. The goal of robust MPC is to take into account uncertainty while planning, whether it be from modeling errors, unmodeled disturbances, or randomness within the system itself [2]. In addition to safety, other considerations such as optimality, real-time tractability, scalability to high dimensional systems, and hard state and control constraints make the problem more difficult. In spite of these difficulties, learning-based robust MPC continues to receive much attention [18, 49, 38, 19, 15, 10, 36, 35, 1, 5]. However, in an effort to satisfy the many competing design requirements in this space, certain restrictive assumptions are often made, which include predetermined error bounds, restricted classes of dynamics models, or fixed parameterizations of the uncertainty.

Consider the following nonlinear dynamics equation that describes a real system:

$$x_{t+1} = f(x_t, u_t) + w_t \tag{1}$$

where $x \in \mathbb{X} \subseteq \mathbb{R}^n$ is the state, $u \in \mathbb{U} \subseteq \mathbb{R}^m$ are controls, and $w \in \mathbb{R}^n$ is noise or disturbance.

When attempting to find a model which captures the behavior of x_t, there will be error that results from insufficient data, lack of knowledge of w_t, or unknown or unobserved higher-dimensional dynamics not observed in x_t. One traditional approach has been to find robust bounds on the model error and plan using this robust model, i.e. $|w_t| \leq W$. However, this approach can be too conservative since it is not time or space varying and does not capture the distribution of the disturbance [1, 42]. To partially address this one could extend W to be time and state-varying, i.e. $W = W(x_t, u_t, t)$, as is commonly done in the robust MPC and control literature. For example, [31] takes this approach for feedback linearizable systems using boundary layer control, [44] leverages contraction theory and sum-of-squares optimization to find stabilizing controllers for nonlinear systems under uncertainty, and [48] solves for forward invariant tubes using min-max differential inequalities (See [25] for a recent overview of other related approaches). In this work we aim to learn this uncertainty directly from data, which allows us to avoid structural assumptions of the system of interest or restrictive parameterizations of uncertainty. We learn a *quantile* representation of the bounds of the distribution of possible trajectories, in the form of a tube around some nominal trajectory (Figure 1).

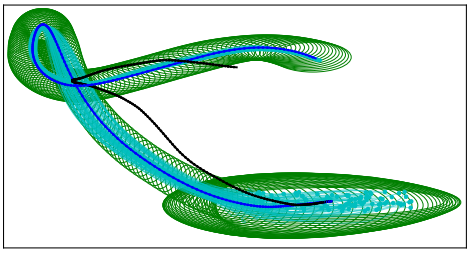


Fig. 1. A learned tube (green) with learned mean (blue) that captures the distribution of trajectories (cyan) on a full quadrotor model tracking a target trajectory (black), propagated for 200 timesteps forward from the initial states (dots).

More closely related to our approach is the wide range of recent work in learning-based planning and control that seeks to handle model uncertainty probabilistically, where a model is constructed from one-step prediction measurements, and it is assumed that the true underlying distribution of the function is Gaussian [11, 20, 8, 30, 3]:

$$P(x_{t+1} | x_t, u_t) = \mathcal{N}(\mu(x_t, u_t), \sigma(x_t, u_t)). \tag{2}$$

where the mean function $\mu : \mathbb{X} \times \mathbb{U} \to \mathbb{X}$ and variance function $\sigma : \mathbb{X} \times \mathbb{U} \to \mathbb{X}^2$ capture the uncertainty of the dynamics for one time step. Various approaches for approximating this posterior distribution have been developed [16, 14]. For example, in PILCO and related work [20], moment matching of the posterior distribution is performed to find an analytic expression for the evolution of the mean and the covariance in time. However, in order to arrive at these analytic expressions, assumptions must be made which lower the descriptive power for the model to capture the true underlying distribution, which

[1]Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA, USA
[2]NASA Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

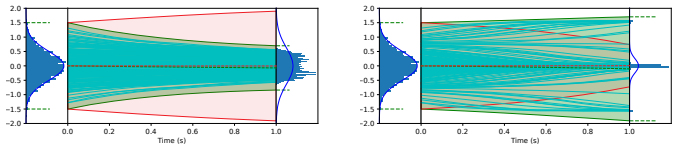


Fig. 2. Comparison of 3-σ bounds on distributions of trajectories using GP moment matching (red) and the proposed quantile regression method (green). 100 sampled trajectories are shown (cyan) along with starting and ending distributions (blue, left and right histograms). Left: GP moment matching overestimates the distribution for the dynamics $\dot{x} = -x|x|$, while our method models it well. Right: GP moment matching underestimates the distribution for the dynamics $\dot{x} = -\sin(4x)$, while our method captures the tails of the distribution.

may be multi-modal and highly non-Gaussian. Furthermore, conservative estimates of the variance of the distribution will grow in an unbounded manner as the number of timesteps increases [26]. The result is that any chance constraints derived from these approximate models may be inaccurate. In Figure 2 we compare the classic GP-based moment matching approach for propagating uncertainty with our own deep quantile regression method on two different functions. While GP-moment matching can both underestimate and overestimate the true distribution of trajectories, our method is less prone to failures due to analytic simplifications or assumptions.

An alternative approach to Bayesian modeling for robust MPC has been to use quantile bounds to bound the tails of the distribution. This has the advantage that for planning in safety-critical contexts, we are generally not concerned with the full distribution of the trajectories, but the tails of these distributions only; specifically, we are interested in the probability of the tail of the distribution violating a safe set. A few recent works have taken this approach in the context of MPC; for example, [4] computes back-off sets with Gaussian Processes, and [5] uses an adaptive control approach to parameterize quantile bounds.

We are specifically interested in the idea of learning quantile bounds using the expressive power of deep neural networks. Quantile bounds give an explicit probability of violation at each timestep and allow for quantifying uncertainty which can be non-Gaussian, skewed, asymmetric, multimodal, and heteroskedastic [46]. Quantile regression itself is a well-studied field with the first results from [23], see also [24, 47]. Quantile regression in deep learning has been also recently considered as a general statistical modeling tool [39, 54, 41, 51, 46]. Bayesian quantile regression has also been studied [27, 52]. Recently quantile regression has gained popularity as a modeling tool within the reinforcement learning community [7].

In addition to introducing a method for deep learning quantile bounds for distributions of trajectories, we also show how this method can be tailored to a tube MPC framework. Tube MPC [28, 33] was introduced as a way to address some of the shortcomings of classic robust MPC; specifically that robust MPC relied on optimizing over an open-loop control sequence, which does not predict the closed-loop behavior well. Instead, tube MPC seeks to optimize over a local policy that generates some closed-loop behavior, which has advantages of robust constraint satisfaction, computational efficiency, and better performance.

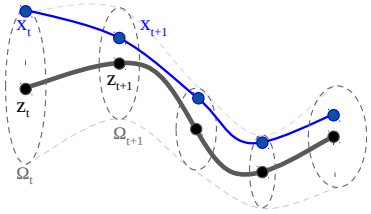


Fig. 3. Diagram of a tube around the dynamics of z, within which x stays invariant. Note that the tube set Ω_t is time-varying.

The use of tube MPC allows us to handle high dimensional systems, as well as making the learning problem more efficient, tractable, and reliable. To the best of our knowledge, our work is the first to combine deep quantile regression with tube-based MPC, or indeed any learning-based robust MPC method.

The structure of the paper is as follows: In Section II we present our approach for learning tubes, which includes deep quantile regression, enforcing a monotonicity condition with a negative divergence loss function, and quantifying epistemic uncertainty. In Section III we present three different learning tube MPC schemes that take advantage of our method. In Section IV we perform several experiments and studies to validate our method, and conclude in Section V.

II. Deep Learning Tubes

A. Learning Tubes For Robust and Tube MPC

We propose learning time-varying invariant sets as a way to address the difficulties with propagating uncertainty for safety critical control, as well as to characterize the performance of a learned model or tracking controller. Consider the following *quantile* description of the dynamics:

$$\begin{aligned} x_{t+1} &= f(x_t, u_t) + w_t \\ z_{t+1} &= f_z(z_t, v_t) \\ \omega_{t+1} &= f_\omega(\omega_t, z_t, v_t, t) \\ P(d(x_t, z_t) \leq \omega_t) &\geq \alpha, \quad \forall t \in \mathbb{N} \end{aligned} \tag{3}$$

where $z \in \mathbb{Z} \subseteq \mathbb{R}^{n_z}$ is a latent state of equal or lower dimension than x, i.e. $n_z \leq n$, and $v \in \mathbb{V} \subseteq \mathbb{R}^{m_z}$ is a pseudo-control input, also of equal or lower dimension than u, i.e. $m_z \leq m$. In the simplest case, we can fix $v_t = u_t$ and/or $z_t = x_t$. Also, $\omega \in \mathbb{R}^{n_z}$ is a vector that we call the *tube width*, with each element of $\omega > 0$.

This defines a "tube" around the trajectory of z within which x will stay close to z with probability greater than $\alpha \in [0,1]$ (Figure 3). More formally, we can define the notion of closeness between some x and z by, for example, the distance between z and the projection of x onto $\mathbb{Z}$: $d(x,z) = |P_{\mathbb{Z}}(x) - z| \in \mathbb{R}^{n_z}$, where $P_{\mathbb{Z}}$ is a projection operator. Let $\Omega_\omega(z) \subset \mathbb{X}$ be a set in $\mathbb{X}$ associated with the tube width ω and z:

$$\Omega_\omega(z) := \{x \in \mathbb{X} : d(x,z) \leq \omega\}. \tag{4}$$

where the $\leq$ is element-wise. Other tube parameterizations are possible, for example $\Omega_\omega(z) := \{x \in \mathbb{X} : \|P_{\mathbb{Z}}(x) - z\|_\omega \leq 1\}$, where $\omega \in \mathbb{R}^{n_z \times n_z}$ instead.

The coupled system (3) induces a sequence of sets $\{\Omega_{\omega_t}(z_t)\}_{t=0}^{T}$ that form a tube around z_t. Our goal is to learn

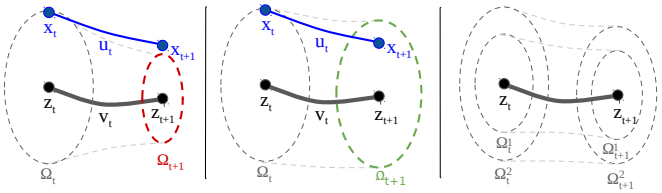


Fig. 4. Learning tube dynamics from data. Left: The predicted tube at $t+1$ is too small. The gradient of the loss function will increase its size. Middle: The predicted tube at $t+1$ is larger than the actual trajectory in x taken, and will be shrunk. Right: The mapping $f_\omega(\omega,z_t,v_t,t)$ is monotonic with respect to ω, which results in $\Omega_t^1 \subseteq \Omega_t^2 \implies \Omega_{t+1}^1 \subseteq \Omega_{t+1}^2$.

how this tube changes over time in order to use it for planning safe trajectories.

B. Quantile Regression

Our challenge is to learn the dynamics of the tube width, f_ω. Given data collected as trajectories $\mathcal{D} = \{x_t, u_t, x_{t+1}, z_t, v_t, z_{t+1}, t\}_{t=0}^T$, we can formulate the learning problem for f_ω as follows.

Let f_ω be parameterized with a neural network, f_ω^θ. For a given t and data point $\{x_t,u_t,x_{t+1},z_t,v_t,z_{t+1},t\}$, let $\omega_t = d(x_t,z_t)$ be the input tube width to f_ω, and $\omega_{t+1} = d(x_{t+1},z_{t+1})$ the candidate output tube width. The candidate tube width at $t+1$ must be less than the estimate of the tube width at $t+1$, i.e: $\omega_{t+1} \leq f_\omega^\theta(\omega_t,z_t,v_t,t)$. To train the network f_ω^θ to respect these bounds we can use the following *check loss* function:

$$L_\omega^\alpha(\theta,\delta) = L^\alpha(\omega_{t+1}, f_\omega^\theta(\omega_t,z_t,v_t,t)) \tag{5}$$
$$L^\alpha(y,r) = \begin{cases} \alpha|y-r| & y > r \\ (1-\alpha)|y-r| & y \leq r \end{cases}$$

where the loss is a function of each data sample $\delta = \{\omega_{t+1},\omega_t,z_t,v_t,t\}$. With the assumption of i.i.d. sampled data, when $L_\omega^\alpha(\theta,\delta)$ is minimized the quantile bound will be satisfied, (see Figure 4 and Theorem II.1). In practice we can smooth this loss function near the inflection point $y=r$ with a slight modification, by multiplying L_ω^α with a Huber loss [21, 7].

Theorem II.1. *Let θ^* minimize $\mathbb{E}_\delta[L_\omega^\alpha(\theta,\delta)]$. Then with probability α, $f_\omega^{\theta^*}(\omega,z,v,t)$ is an upper bound for $f_\omega(\omega,z,v,t)$.*

Proof. With a slight abuse of notation, let x denote the input variable to the loss function, and consider the expected loss $\mathbb{E}_x[L^\alpha(y(x),r(x))]$. We find the minimum of this loss w.r.t. r by setting the gradient to 0:

$$\begin{aligned} &\frac{\partial}{\partial r^*}\mathbb{E}_x[L^\alpha(y(x),r^*(x))] \\ &= \int_{y(x)>r^*(x)} \alpha p(x)dx - \int_{y(x)\leq r^*(x)} (1-\alpha)p(x)dx \\ &= \alpha p(y(x)>r^*(x)) - (1-\alpha)p(y(x)\leq r^*(x)) = 0 \\ &\implies p(y(x)\leq r^*(x)) = \alpha \end{aligned} \tag{6}$$

Replacing $r^*(x)$ with $f_\omega^{\theta^*}(\omega,z,v,t)$ and $y(x)$ with $f_\omega(\omega,z,v,t)$ completes the proof. □

Note that quantile regression gives us tools for learning tube dynamics $f_\omega(\omega, z, v, t, \alpha)$ that are a function of the quantile probability α as well. This opens the possibility to dynamically varying the margin of safety while planning, taking into account acceptable risks or value at risk [12]. For example, in planning a trajectory, one could choose a higher α for the near-term and lower α in the later parts of the trajectory, reducing the conservativeness of the solution.

Additionally, we note that we can train the tube bounds dynamics in a recurrent fashion to improve long sequence prediction accuracy. While we present the above and following theorems in the context of one timestep, they are easily extensible to the recurrent case.

C. Enforcing Monotonicity

In addition to the quantile loss we also introduce an approach to enforce monotonicity of the tube with respect to the tube width (Figure 4, right). This is important for ensuring recursive feasibility of the MPC problem, as well as allowing us to shrink the tube width during MPC at each timestep if we obtain measurement updates of the current state, or, in the context of state estimation, an update to the covariance of the estimate of the current state. Enforcing monotonicity in neural networks has been studied with a variety of techniques [43, 53]. Here we adopt the approach of using a loss function that penalizes the network for having negative divergence, similar to [17]:

$$L_m(\theta,\delta) = -\min(0, \text{div}_\omega f_\omega(\omega,z,v,t)) \tag{7}$$

where div_ω is the divergence of f_ω with respect to ω. In practice we find that under gradient-based optimization, this loss decreases to 0 in the first epoch and does not noticeably affect the minimization of the quantile loss. Minimizing $L_m(\theta,\delta)$ allows us to make claims about the monotonicity of the learned tube:

Theorem II.2. *Suppose θ^* minimizes $\mathbb{E}_\delta[L_m(\theta,\delta)]$ and $\mathbb{E}_\delta[L_m(\theta^*,\delta)] = 0$. Then for any $z_t \in \mathbb{Z}$, $v_t \in \mathbb{V}$, $t \in \mathbb{N}$ and ω_t^1, $\omega_t^2 \in \mathbb{R}^{n_z}$, if $\Omega_{\omega_t^1}(z_t) \subseteq \Omega_{\omega_t^2}(z_t)$, then $\Omega_{\omega_{t+1}^1}(z_{t+1}) \subseteq \Omega_{\omega_{t+1}^2}(z_{t+1})$.*

Proof. Since $\forall\theta,\delta,\ L_m(\theta,\delta) > 0$ and $\mathbb{E}[L_m(\theta^*,\delta)] = 0$, then $L_m(\theta^*,\delta) = 0$. Then $\nabla_\omega f_\omega(\omega,z,v,t) > 0$ and f_ω is nondecreasing with respect to ω. Since $\Omega_{\omega_t^1} \subseteq \Omega_{\omega_t^2}$, then $\omega_t^1 \leq \omega_t^2$, so $f_\omega(\omega_t^1,z_t,v_t,t) \leq f_\omega(\omega_t^2,z_t,v_t,t)$, which implies that $\Omega_{\omega_{t+1}^1}(z_{t+1}) \subseteq \Omega_{\omega_{t+1}^2}(z_{t+1})$. □

D. Epistemic Uncertainty

Finally, in order to account for uncertainty in regions where no data is available for estimating quantile bounds, we incorporate methods for estimating epistemic uncertainty. Such methods can include Bayesian neural networks, Gaussian Processes, or other heuristic methods in deep learning [13, 7, 37]. For the experiments in this work we adopt an approach that adds an additional output layer to our quantile regression network that is linear with respect to orthonormal weights [46]. We emphasize that a wide range of methods for quantifying epistemic uncertainty are available and we are not restricted to this one approach; however, for the sake of clarity, we present in detail our method of choice. Let $g(z,v,t)$ be a neural network with either fixed weights that are either

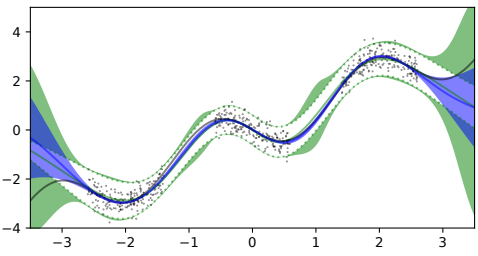


Fig. 5. Estimating epistemic uncertainty for a 1-D function. Black dots indicate noisy data used to train the models, black line indicates the true function. Green colors indicate trained neural network models with green line indicating mean, and green dotted lines indicating learned 99% quantile bounds. Green shading indicates increased quantile bounds scaled by the learned epistemic uncertainty. Blue line and shading is GP regression with 99% bounds for comparison.

randomly chosen or pre-trained, with l dimensional output. We branch off a second output with a linear layer: $C^\intercal g(z,v,t)$, where $C \in \mathbb{R}^{l\times k}$. The estimate of epistemic uncertainty is chosen as $u_e(z,v,t) = \|C^\intercal g(z,v,t)\|^2$. Then, the parameters C are trained by minimizing the following loss:

$$L_u(C,\delta) = \|C^\intercal g(z,v,t)\|^2 + \lambda\|C^\intercal C - I_k\|. \tag{8}$$

where $\lambda > 0$ weights the orthonormal regularization. Minimizing this loss produces a network that has a value close to 0 when the input data is in-distribution, and increases with known rate as the input data moves farther from the training distribution (Figure 5, and see [46] for detailed analysis). We scale the predicted quantile bound by the epistemic uncertainty, then add a maximum bound to prevent unbounded growth as ω grows:

$$f_\omega(\omega,z,v,t) \leftarrow \min\{(1+\beta u_e(z,v,t)) f_\omega(\omega,z,v,t), W\} \tag{9}$$

where $\beta > 0$ is a constant parameter that scales the effect of the epistemic uncertainty, and W is a vector that provides an upper bound on the total uncertainty. Finding an optimal β analytically may require some assumptions such as a known Lipschitz constant of the underlying function, non-heteroskedastic noise, etc., which we leave for future investigation. We set β and W by hand and find this approach to be effective in practice.

We expect that as the field matures, methods for providing guarantees on well-calibrated epistemic uncertainty in deep learning will continue to improve. In the meantime, we make the assumption that we have well-calibrated epistemic uncertainty, an assumption similar to those made with other learning-based controls methods, such as choosing noise covariances, disturbance magnitudes, or kernel types and widths. The main benefit of leveraging epistemic uncertainty modeling is that it allows us to maintain guarantees of safety and recursive feasibility when we have a limited amount of data to learn from. In the case when no reliable epistemic estimate is available, we can proceed if we simply assume there is sufficient data to learn a good model offline.

III. Three Ways to Learn Tubes for Tube MPC

In this section we present three variations for applying our deep quantile regression approach to MPC problems, whose applicability may vary based on what components are available to the designer. By leveraging the previously described theorems for ensuring accurate quantiles, monotonicity, and uncertainty of the tube width dynamics, we can guarantee recursive feasibility of these MPC schemes, while ensuring that the trajectory of the system x_t remains within a safe set $x_t \in \mathcal{C} \subset \mathbb{R}^n$ with probability α at each timestep. The three different approaches require different elements of the system to be known or given, and are summarized as:

1) Given a tracking control law $u = \pi(x,z)$ and reference trajectory dynamics f_z, construct an invariant tube with the reference trajectory at its center (Figure 3).
2) Given a tracking control law π and reference trajectory dynamics f_z, construct a model of the dynamics of the error $e = x - z$, then learn an invariant tube with $z+e$ as its center (Figure 6a).
3) From data generated from any control law, random or otherwise, learn a reduced representation of the dynamics f_z (and optionally, a policy π to track it), along with tube bounds on the tracking error (Figure 6b).

A. Learning Tube Dynamics for a Given Controller

We first consider the case where we are given a fixed ancillary controller $\pi(x,z) : \mathbb{X}\times\mathbb{Z} \to \mathbb{U}$ (or potentially $\pi(x,z,v)$ with a feed-forward term v), along with nominal dynamics f_z that are used for planning and tracking in the classic tube MPC manner [32]. For now our goal is to learn f_ω alone.

We sum the three losses discussed in the previous section:

$$L(\theta,C,\delta) = L^\alpha_\omega(\theta,\delta) + L_m(\theta,\delta) + L_u(C,\delta) \tag{10}$$

to learn f^θ_ω, and find θ^* and C^* via stochastic gradient descent. Next, we perform planning on the coupled z and tube dynamics in the following nonlinear MPC problem. Let $T \in \mathbb{N}$ denote the planning horizon. We use the subscript notation $v_{k|t}$ to denote the variable v_k for $k = 0,\cdots,T$ within the MPC problem at time t. Let $v_{\cdot|t}$ denote the set of variables $\{v_{k|t}\}_{k=0}^T$. Then, at time t, the MPC problem is:

$$\min_{v_{\cdot|t}\in\mathbb{V}} J_T(v_{\cdot|t}, z_{\cdot|t}, \omega_{\cdot|t}) \tag{11a}$$

$$s.t. \forall k = 0,\cdots,T:$$

$$z_{k+1|t} = f_z(z_{k|t}, v_{k|t}) \tag{11b}$$

$$\omega_{k+1|t} = f^\theta_\omega(\omega_t, z_t, v_t, t) \tag{11c}$$

$$\omega_{0|t} = d(x_t, z_{0|t}) \tag{11d}$$

$$z_{T|t} = f_z(z_{T|t}, v_{T|t}) \tag{11e}$$

$$\omega_{T|t} \geq f^\theta_\omega(\omega_{T|t}, z_{T|t}, v_{T|t}, T) \tag{11f}$$

$$\Omega_{\omega_{k|t}}(z_{k|t}) \subseteq \mathcal{C} \tag{11g}$$

Let $v^*_{\cdot|t}, z^*_{\cdot|t}$ denote the minimizer of the problem at time t. Note that we include $\omega_{\cdot|t}$ in the cost, which allows us to encourage larger or smaller tube widths. The tube width $\omega_{0|t}$ is updated based on a measurement x_t from the system, or can also be updated with information from a state estimator. In the absence of measurements we can also carry over the past optimized tube width, i.e. $\omega_{0|t} = \omega^*_{1|t-1}$, as long as $x_t \in \Omega_{\omega_{0|t}}(z_{0|t})$. The closed-loop control is set to $v_t = v^*_{0|t}$

Algorithm 1: Tube Learning for Tube MPC

Require: Ancillary policy π, Latent dynamics f_z, Safe set $\mathcal{C}$, Quantile probability α. MPC horizon T.
Initialize: Neural network for tube dynamics f_ω^θ. Dataset $\mathcal{D}=\{x_{t_i},u_{t_i},x_{t_i+1},z_{t_i},v_{t_i},z_{t_i+1},t_i\}_{i=1}^N$. Initial states x_0, z_0, Initial feasible controls $v_{\cdot|0}$.
for $t=0,\cdots$ **do**
  if *updateModel* **then**
    Train f_ω^θ on dataset $\mathcal{D}$ by minimizing tube dynamics loss (10).
  if x_t *measured* **then**
    Initialize tube width $\omega_{0|t}=d(x_t,z_t)$
  Solve MPC problem (11) with warm-start $v_{\cdot|t}$, obtain v_t, z_{t+1}
  Apply control policy to system $u_t=\pi(x_t,z_{t+1})$
  Step forward for next iteration: $v_{k|t+1}=v^*_{k+1|t}$, $k=0,\cdots,T-1$, $v_{T|t+1}=v^*_{T|t}$, $z_{0|t+1}=z^*_{1|t}$, $\omega_{0|t+1}=\omega^*_{1|t}$
  Append data to dataset $\mathcal{D}\leftarrow\mathcal{D}\cup\{x_t,u_t,x_{t+1},z_t,v_t,z_{t+1},t\}$

and the tracking target for the underlying policy is $z_{t+1}=z^*_{1|t}$. Under these assumptions we have the following theorem establishing recursive feasibility and safety:

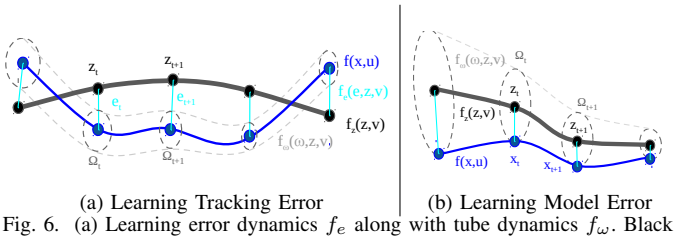
Theorem III.1. *Suppose that the MPC problem (11) is feasible at $t=0$. Then the problem is feasible for all $t>0\in\mathbb{N}$ and at each timestep the constraints are satisfied with probability α.*

Proof. The proof is similar to that in [25] for general set-based robust adaptive MPC. Let $z_{0|t+1}=z^*_{1|t}$ and choose any $\omega_{0|t+1}$ such that $x_{t+1}\in\Omega_{\omega_{0|t+1}}(z_{0|t+1})$ (if measurements x_{t+1} are unavailable, one can use $\omega_{0|t+1}=\omega^*_{1|t}$). With probability α, $\Omega_{\omega_{0|t+1}}(z_{0|t+1})\subseteq\Omega_{\omega^*_{1|t}}(z^*_{1|t})$ due to Theorem II.1. Let $v_{k|t+1}=v^*_{k+1|t}$ for $k=0,\cdots,T-1$, and let $v_{T|t+1}=v^*_{T|t}$. Then $v_{\cdot|t+1}$ is a feasible solution for the MPC problem at $t=1$, due to the terminal constraints (11e,11f) as well as the monotonicity of f_ω with respect to ω (Theorem II.2). □

Since $f_\omega^\theta(\omega_t,z_t,v_t)$ is nonlinear we find solutions to the MPC problem via iterative linear approximations, yielding an SQP MPC approach [9, 6]. Other optimization techniques are possible, including GPU-accelerated sampling-based ones [50]. We outline the entire procedure in Algorithm 1.

B. Learning Tracking Error Dynamics and Tube Dynamics

Next we show how to learn error dynamics $e_{t+1}=f_e(e_t,z_t,v_t)$ along with a tube centered along these dynamics, where $e_t=P_\mathbb{Z}(x)-z$ is the error between x and z, with x projected onto $\mathbb{Z}$. These error dynamics function as the mean of the distribution of dynamics $x_{t+1}=f(x_t,u_t)$ when the tracking policy is used $u_t=\pi(x_t,z_{t+1},v_t)$. This allows the tube to take on a more accurately parameterized shape (Figure 6a). Setting up the learning problem in this way offers several distinct advantages. First, rather than relying on an accurate nominal model f_z and learning the bounds between this model and the true dynamics, we directly characterize the difference between the two models with f_e. This means that f_z can be chosen more arbitrarily and does not need to be a high-fidelity dynamics



(a) Learning Tracking Error (b) Learning Model Error

Fig. 6. (a) Learning error dynamics f_e along with tube dynamics f_ω. Black line is the nominal trajectory f_z, blue line is data collected from the system. Cyan indicates tracking errors, whose dynamics are learned. Grey tube denotes f_ω, which captures the error between the true dynamics and z_t+e_t. (b) Fitting learned dynamics to actual data. Blue inline indicates data collected from the system, black line is a learned dynamics trajectory fitted to the data.

model. Second, using the nominal dynamics z_t as an input to f_e and learning the error "anchors" our prediction of the behavior of x_t to z_t. This allows us to predict the expected distribution of x_t with much higher accuracy for long time horizons, in contrast to the approach of learning a model f directly and propagating it forward in time, where the error between the learned model and the true dynamics tends to increase with time.

As before, we assume we have a known π and nominal dynamics f_z. Let $\Omega^e_\omega(z,e)\subset\mathbb{X}$ be a set in $\mathbb{X}$ associated with the tube width ω,z, and e:

$$\Omega^e_\omega(z,e):=\{x\in\mathbb{X}:d(x,z+e)\leq\omega\}. \tag{12}$$

where the $\leq$ is element-wise. We have the following description of the error dynamics:

$$\begin{aligned} e_{t+1}&=f_e(e_t,z_t,v_t) \\ \omega_{t+1}&=f_\omega(\omega_t,z_t,v_t) \\ P(|(z_t+e_t)+x_t|\leq\omega_t)&\geq\alpha, \qquad \forall t\in\mathbb{N} \end{aligned} \tag{13}$$

Given a dataset $\mathcal{D}=\{x_t,\,u_t,\,x_{t+1},\,z_t,\,v_t,\,z_{t+1},\,t\}_{t=0}^N$, we minimize the following loss over data samples $\delta=\{x_t,x_{t+1},z_t,z_{t+1},v_t\}$ in order to learn $f_e(e_t,z_t,v_t)$, which we parameterize with ξ:

$$L_e(\xi,\delta)=\|f_e^\xi(P_\mathbb{Z}(x_t)-z_t,v_t)-P_\mathbb{Z}(x_{t+1})-z_{t+1}\|_2 \tag{14}$$

Next, we learn f_ω by minimizing the quantile loss (10). However, while in the previous section $\omega_t=d(x_t,z_t)$, here we approximate the tube width with $\omega_t=d(x_t,z_t+e_t)$. We obtain e_t by propagating the learned dynamics f_e^ξ forward in time, given z_t,v_t. Then we can solve a similar tube-based robust MPC problem (15):

$$\min_{v_{\cdot|t}\in\mathbb{V}} J_T(v_{\cdot|t},z_{\cdot|t}+e_{\cdot|t},\omega_{\cdot|t}) \tag{15a}$$

$$s.t.\forall k=0,\cdots,T:$$

$$z_{k+1|t}=f_z(z_{k|t},v_{k|t}) \tag{15b}$$

$$e_{k+1|t}=f_e^\xi(e_t,z_t,v_t) \tag{15c}$$

$$\omega_{k+1|t}=f_\omega^\theta(e_t,z_t,v_t,t) \tag{15d}$$

$$\omega_{0|t}=d(x_t,z_{0|t}+e_{0|t}) \tag{15e}$$

$$z_{T|t}+e_{T|t}=f_z(z_{T|t},v_{T|t})+f_e^\xi(e_{T|t},z_{T|t},v_{T|t}) \tag{15f}$$

$$\omega_{T|t}\geq f_\omega^\theta(\omega_{T|t},z_{T|t},v_{T|t},T) \tag{15g}$$

$$\Omega^e_{\omega_{k|t}}(z_{k|t})\subseteq\mathcal{C} \tag{15h}$$

Algorithm 2: Learning Tracking Error Dynamics and Tube Dynamics for Tube MPC

Require: Ancillary policy π, Latent dynamics f_z, Safe set $\mathcal{C}$, Quantile probability α. MPC horizon T.
Initialize: Neural network for error dynamics f_e^ξ. Neural network for tube dynamics f_ω^θ. Dataset $\mathcal{D}=\{x_{t_i},u_{t_i},x_{t_i+1},z_{t_i},v_{t_i},z_{t_i+1},t_i\}_{i=1}^N$. Initial states x_0, z_0, e_0, Initial feasible controls $v_{\cdot|0}$.
for $t=0,\cdots$ **do**
  if *updateModels* **then**
    Train f_e^ξ on dataset $\mathcal{D}$ by minimizing error dynamics loss (14).
    Forward propagate learned model f_x^ξ on dataset $\mathcal{D}$ to obtain $\{e_{t_i}\}_{t=1}^N$. Append to $\mathcal{D}$.
    Train f_ω^θ on dataset $\mathcal{D}$ by minimizing tube dynamics loss (10), but replace $\omega_{t_i}=d(x_{t_i},x_{t_i}+e_{t_i})$.
  if x_t *measured* **then**
    Initialize tube width $\omega_{0|t}=d(x_t,z_t+e_t)$
  Solve MPC problem (15) with warm-start $v_{\cdot|t}$, obtain v_t, z_{t+1}
  Apply control policy to system $u_t=\pi(x_t,z_{t+1},v_t)$
  Step forward for next iteration: $v_{k|t+1}=v^*_{k+1|t}$, $k=0,\cdots,T-1$, $v_{T|t+1}=v^*_{T|t}$, $z_{0|t+1}=z^*_{1|t}$, $e_{0|t+1}=e^*_{1|t}$, $\omega_{0|t+1}=\omega^*_{1|t}$
  Append data to dataset $\mathcal{D}\leftarrow\mathcal{D}\cup\{x_t,u_t,x_{t+1},z_t,v_t,z_{t+1},t\}$

Notice that the cost and constraints are now a function of z_t+e_t and do not depend on z_t only. This means that we are free to find paths z_t for the tracking controller π to track, which may violate constraints. We maintain the same guarantees of feasibility and constraint satisfaction as in Theorem III.1. Since the proof is similar we omit it for brevity. See Algorithm 2.

Theorem III.2. *Suppose that the MPC problem (15) is feasible at $t=0$. Then the problem is feasible for all $t>0\in\mathbb{N}$ and at each timestep the constraints are satisfied with probability α.*

C. Learning System Dynamics and Tube Dynamics

In our third approach to learning tubes, we wish to learn the dynamics directly without a prior nominal model f_z. We restrict $\mathbb{Z}=\mathbb{X}$ and $\mathbb{V}=\mathbb{U}$, and treat z as an approximation of x. Our goal is to learn f_z to approximate f, along with f_ω that will determine a time-varying upper bound on the model error. Typically the open-loop model error will increase in time in an unbounded manner, which may make it difficult to find a feasible solution to the MPC problem. One approach is to assume the existence of a stabilizing controller and terminal set, and use a terminal condition that ensures the trajectory ends in this set [22, 26]. A second approach is to find a feedback control law π to ensure bounded tube widths. We describe the latter approach in more detail, but do not restrict ourselves to it.

Using a standard L2 loss function, we first learn an approximation of f, call it f_z^ϕ with parameters ϕ:

$$L_f(\phi,\delta)=\|f_z^\phi(x_t,u_t)-x_{t+1}\|_2 \tag{16}$$

Next, we learn a policy π^ψ with parameters ψ by inverting the dynamics:

$$L_\pi(\psi,\delta)=\|\pi^\psi(x_t,x_{t+1})-u_t\|_2 \tag{17}$$

By learning a policy in this manner we decouple the potentially inaccurate model $f_z^\phi(x_t,u_t)$ from the true dynamics, in a learning inverse dynamics fashion [34]. To see this, suppose we have some z_t and v_t, and $z_{t+1}=f_z^\phi(z_t,v_t)$. If $x_t\neq z_t$ and we apply v_t to the real system, $x_{t+1}=f(x_t,v_t)$, then the error $\|x_{t+1}-z_{t+1}\|$ will grow, i.e. $\|x_t-z_t\|\leq\|x_{t+1}-z_{t+1}\|$. However, if instead we use the policy π^ψ, then $f(x_t,\pi^\psi(x_t,z_{t+1}))$ should be closer to z_{t+1}, and the error is more likely to shrink. Other approaches are available for learning π, including reinforcement learning [45], imitation learning [40], etc. Finally, we learn f_ω in the same manner as before by minimizing the quantile loss in (10). We generate data for learning the tube dynamics by fitting trajectories of the learned model f_z^ϕ to closely approximate the real data x_t (Figure 6b). We randomly initialize $z_{0|t}$ along the trajectory x_t by letting $z_{0|t}=\mathcal{N}(x_t,\sigma I)$. We solve the following problem for each t:

$$\min_{v_{\cdot|t}\in\mathbb{V}}\sum_{k=1}^{T}\|z_{k|t}-x_{t+k}\| \tag{18a}$$

$$s.t.\quad z_{k+1|t}=f_z^\phi(z_{k|t},v_{k|t}),\quad \forall k=0,\cdots,T-1 \tag{18b}$$

From the fitted dynamics model data, we collect tube training data $\mathcal{D}_z=\bigcup_t\left[\{x_{t+k},x_{t+k+1},z_{k|t},v_{k|t},z_{k+1|t}\}_{k=0}^T\right]$ and proceed to train the tube model. We can now solve the same tube-based robust MPC problem (11), with f_z replaced with f_z^ϕ. This allows us to maintain the same guarantees of feasibility and safety with probability α as before. See Algorithm 3.

IV. Experimental Details

A. Evaluation on a 6-D problem

In this section we validate each of our three approaches to learned tubes for tube MPC on a 6-state simulated triple-integrator system. We introduce two sets of dynamics for f and f_z to demonstrate our method. Consider the following 2D triple-integrator system with 6 states, where $x=[p_x,p_y,v_x,v_y,a_x,a_y]^\intercal$, along with the 4 state 2D double-integrator dynamics for the reference system: $z=[p_x^z,p_y^z,v_x^z,v_y^z]$. Let these systems have the following dynamics (we show the x-axis only for brevity sake):

$$\frac{d}{dt}\begin{bmatrix}p_x\\v_x\\a_x\end{bmatrix}=\begin{bmatrix}0&1&0\\0&0&1\\0&0&-k_f\end{bmatrix}\begin{bmatrix}p_x\\v_x\\a_x\end{bmatrix}+\begin{bmatrix}0\\0\\1\end{bmatrix}u_x+\begin{bmatrix}0&0\\1&0\\0&1\end{bmatrix}w \tag{19}$$

$$\frac{d}{dt}\begin{bmatrix}p_x^z\\v_x^z\end{bmatrix}=\begin{bmatrix}0&1\\0&-k_f^z\end{bmatrix}\begin{bmatrix}p_x^z\\v_x^z\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}v_x \tag{20}$$

where $w\sim\mathcal{N}(0,\epsilon I_{2\times2})$, and with similar dynamics for the y-axis. We construct the following cascaded PD control law:

$$\pi_x(p_x,p_x^z)=k_d(k_p(p_x^z-p_x)-v_x+v_x^z)+k_a(-a_x) \tag{21}$$

We choose $k_f=0.1,k_f^z=1.0,k_p=1,k_d=10,k_a=5$, and $\epsilon=0.05$. We also bound $\|v_x\|,\|v_y\|\leq1$. We simulate in discrete time with $dt=0.1$.

Algorithm 3: Learning Dynamics and Model Error Bounds for Tube MPC

Require: Safe set $\mathcal{C}$, Quantile probability α. MPC horizon T.
Initialize: Neural network for policy π^{ψ}, dynamics f_z^{ϕ}, and tube dynamics f_{ω}^{θ}. Dataset $\mathcal{D}=\{x_{t_i},u_{t_i},x_{t_i+1}\}_{i=1}^{N}$. Initial state x_0.
Solve MPC problem (11) for initial feasible control sequence $v_{\cdot|0}$.
for $t=0,\cdots$ **do**
  if *updateModel* **then**
    Train f_z^{ϕ} on dataset $\mathcal{D}$ by minimizing dynamics loss (16).
    Train π^{ψ} on dataset $\mathcal{D}$ by minimizing policy loss (17).
    Create $\mathcal{D}_z=\bigcup_t\left[\{x_{t+k},x_{t+k+1},z_{k|t},v_{k|t},z_{k+1|t}\}_{k=0}^{T}\right]$ by solving (18).
    Train f_{ω}^{θ} on dataset $\mathcal{D}_z$ by minimizing tube dynamics loss (10).
  if x_t *measured* **then**
    Initialize tube width $\omega_{0|t}=d(x_t,z_t)$
  Solve MPC problem (11) with warm-start $v_{\cdot|t}$, obtain v_t, z_{t+1}
  Apply control policy to system $u_t=\pi^{\psi}(x_t,z_{t+1})$
  Step forward for next iteration: $v_{k|t+1}=v^*_{k+1|t}$, $k=0,\cdots,T-1$, $v_{T|t+1}=v^*_{T|t}$, $z_{0|t+1}=z^*_{1|t}$, $\omega_{0|t+1}=\omega^*_{1|t}$
  Append data to dataset $\mathcal{D}\leftarrow\mathcal{D}\cup\{x_t,u_t,x_{t+1}\}$

We collect $\sim$100 episodes with randomly generated controls, with episode lengths of $\sim$100 steps. Following each algorithm, we then set up an MPC task to navigate through a forest of obstacles (see Figure 7). We found an MPC planning horizon of 20-30 steps to be effective. We ran each MPC algorithm for 100 steps, or until the system reaches the goal. We also plot 100 rollouts of the "true" system x_t to evaluate the learned bounds. For each learned network, we use 3 layers with 256 units each. When calculating constraints for the tube, we treat the tube width ω_t as axes for an ellipse rather than a box. This alleviates the need for solving a mixed integer quadratic program, at the cost of a slightly larger tube. We use a quadratic running cost that penalizes deviation from the goal and excessively large velocities.

With Algorithm 1, we note that the tube widths are quite large. This is because this algorithm uses the reference trajectory itself as the center of the tube. While the tube encloses the trajectories, it does not create a tight bound. In Algorithm 2, we address this issue directly. We learn dynamics of the mean tracking error and use this as our tube center. The resulting tube dynamics bound the state distribution more closely. Note that when solving the MPC problem, the optimized reference trajectory z_t is free to violate the constraints, as long as the system trajectories x_t do not. This approach allows for much more aggressive behaviors. For Algorithm 3, without a good tracking controller, the tube width increases over time. However, because we replan at each timestep with a finite horizon, the planner is still able to fit through narrow passages. In the example shown we replan from the current state x_t, with the assumption that it is measured. This allows us to create aggressive trajectories with narrow tube widths.

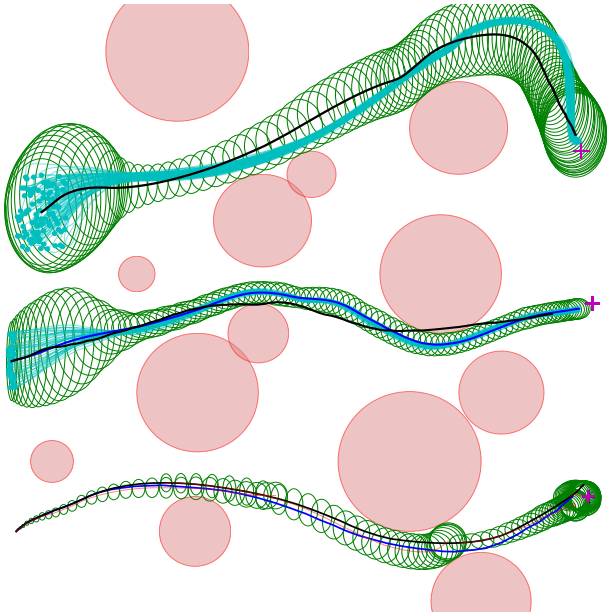


Fig. 7. Comparison of 3 tube MPC approaches with learned tubes. Red circles denote obstacles, magenta cross denotes goal. Cyan lines indicate sampled trajectories from the system x_t with randomized initial conditions. Top: Algorithm 1, learning a tube around the reference z (black) used for tracking. Green circles indicate the tube width obtained at each timestep. Mid: Algorithm 2, learning tracking error dynamics (blue line) for the center of the tube. Bot: Algorithm 3, tube MPC problem using learned policy, dynamics, and tube dynamics. Red lines indicate planned NN dynamics trajectories at each MPC timestep, along with the forward propagated tube dynamics (green), shown every 20 timesteps. Blue line indicates actual path taken (x_t).

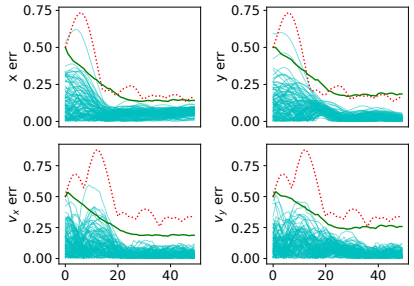


Fig. 8. Learned 95% quantile error bounds (green) vs. 95% analytic bounds (dotted red) for the linear triple-integrator system, with 100 sampled trajectories, tracking a random reference trajectory.

B. Comparison with analytic bounds

We compare our learned tubes with an analytic solution for robust bounds on the system (20). We derive these analytic bounds by assuming worst-case noise perturbations of the closed-loop system. We find the bound W such that $P(|w_t|\leq W)\geq\alpha$ (with $\alpha=0.95$). The worst-case error at each timestep is $w_t=\pm W$. We compare these bounds with those learned with our quantile method (Figure 8). Our method tends to underestimate the true bounds slightly, which is due to the training data rarely containing worst-case adversarial noise sequences.

C. Ablative Study

We perform an ablative study of our tube learning method. Using Algorithm 1, we learn error dynamics and tube dynamics.

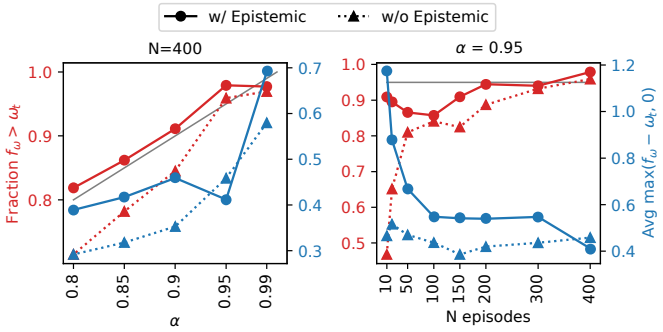


Fig. 9. Evaluation of learned tube dynamics f_ω on triple integrator system with varying α (left) and varying number of datapoints (right). Red indicates fraction of validation samples that exceed the bound, while blue indicates average distance in excess of the bound. Models learned with the epistemic loss along with the quantile loss (circles, solid lines) perform better vs. models without epistemic uncertainty (triangles, dotted lines). Gray lines mark the best possible values.

We collect randomized data (400 episodes of 40 timesteps) and train f_ω under varying values of α. We then evaluate the accuracy of f_ω by sampling 100 new episodes of 10 timesteps, and plot the frequency that f_ω overestimates the true error, along with the magnitude of overestimation (Figure 9, left). We compare networks learned with the epistemic loss and without it, and find that our method produces well-calibrated uncertainties when using the epistemic loss, along with the quantile and monotonic losses (10). We evaluated ablation of the monotonic loss but found no noticeable differences.

We also evaluate estimation of epistemic uncertainty with varying amounts of data (from 10 to 400 episodes), with a fixed value of $\alpha = 0.95$. We find that estimating epistemic uncertainty is particularly helpful in the low-data regimes (Figure 9, right). As expected, the network maintains good quantile estimates by increasing the value of f_ω, which results in larger tubes. This creates more conservative behavior when the model encounters new situations.

D. Evaluation on Quadrotor Dynamics

To validate our approach scales well to high-dimensional non-linear systems, we apply Algorithm 2 to a 12 state, 4 input quadrotor model, with dynamics:

$$\dot{\mathbf{x}} = \mathbf{v} \qquad m\dot{\mathbf{v}} = mg\mathbf{e}_3 - TR\mathbf{e}_3$$
$$\dot{R} = R\hat{\Omega} \qquad J\dot{\Omega} = M + w - \Omega \times J\Omega$$

where $\hat{\cdot} : \mathbb{R}^3 \to SO(3)$ is the hat operator. The states are the position $\mathbf{x} \in \mathbb{R}^3$, the translational velocity $\mathbf{v} \in \mathbb{R}^3$, the rotation matrix from body to inertial frame $R \in SO(3)$, and the angular velocity in the body frame $\Omega \in \mathbb{R}^3$. $m \in \mathbb{R}$ is the mass of the quadrotor, $g \in \mathbb{R}$ denotes gravitational force, and $J \in \mathbb{R}^{3\times3}$ is the inertia matrix in body frame. The inputs to the model are the total thrust $T \in \mathbb{R}$ and the total moment in the body frame $M \in \mathbb{R}^3$. Noise enters through the control channels, with $w \sim \mathcal{N}(0, \epsilon I_{3\times3})$. Our state is $x_t = \{\mathbf{x}, \mathbf{v}, R, \Omega\} \in \mathbb{R}^{18}$ and control input is $u_t = \{T, M\} \in \mathbb{R}^4$. We use a nonlinear geometric tracking controller that consists of a PD controller on position and velocity, which then cascades to an attitude controller [29]. For the nominal model f_z we use a double integrator system on each position axis. The nominal state is $z_t = \{\mathbf{x}, \mathbf{v}\} \in \mathbb{R}^6$ with acceleration control inputs $v_t = \{a_x, a_y, a_z\} \in \mathbb{R}^3$. See Figures 10 and 11.

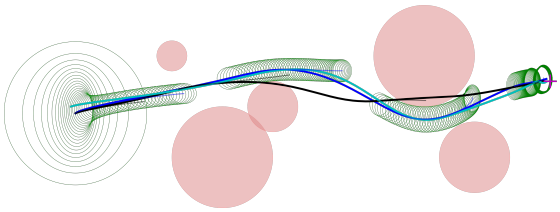


Fig. 10. Algorithm 2 working on quadrotor dynamics, showing 5 individual MPC solutions at different times along the path taken. Thinner lines (black and blue) indicate planned future trajectories $z_{\cdot|t}$ and $e_{\cdot|t}$, respectively.

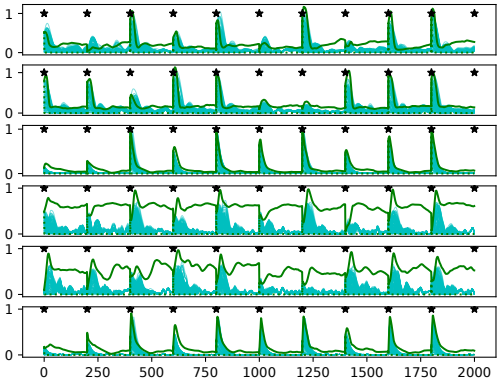


Fig. 11. Tube widths f_ω for quadrotor dynamics, 10 episodes of 200 timesteps each, tracking random reference trajectories. From top to bottom, we plot $(p_x, p_y, p_z, v_x, v_y, v_z)$. Green lines indicate the quantile bound ω_t, with $\alpha = 0.9$, and cyan lines show 100 sampled error trajectories, $|x_t - e_t|$. Black stars indicate the start of a new episode.

V. Conclusion

We have introduced a deep quantile regression framework for learning bounds on controlled distributions of trajectories. For the first time we combine deep quantile regression in three robust MPC schemes with recursive feasibility and constraint satisfaction guarantees. We show that these schemes are useful for high dimensional learning-based control on quadrotor dynamics. We hope this work paves the way for more detailed investigation into a variety of topics, including deep quantile regression, learning invariant sets for control, handling epistemic uncertainty, and learning-based control for non-holonomic or non-feedback linearizable systems. Our immediate future work will involve hardware implementation and evaluation of these algorithms on a variety of systems.

Acknowledgement

We thank Brett Lopez and Rohan Thakker for insightful discussions and suggestions, as well as the reviewers for helpful comments. This research was partially carried out at the Jet Propulsion Laboratory (JPL), California Institute of Technology, and was sponsored by the JPL Year Round Internship Program and the National Aeronautics and Space Administration (NASA).

References

[1] Anil Aswani, Humberto Gonzalez, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, may 2013.

[2] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In *Robustness in identification and control*, pages 207–226. Springer, 1999.

[3] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe Model-based Reinforcement Learning with Stability Guarantees. In *Advances in neural information processing systems*, pages 908–918, 2017.

[4] Eric Bradford, Lars Imsland, and Ehecatl Antonio del Rio-Chanona. Nonlinear model predictive control with explicit back-offs for gaussian process state space models. In *58th Conference on decision and control (CDC). IEEE*, 2019.

[5] Monimoy Bujarbaruah, Xiaojing Zhang, Marko Tanaskovic, and Francesco Borrelli. Adaptive MPC under time varying uncertainty: Robust and Stochastic. *arXiv preprint arXiv:1909.13473*, 2019.

[6] Rui Camacho, Ross King, and Ashwin Srinivasan. *Inductive Logic Programming: 14th International Conference, ILP 2004, Porto, Portugal, September 6-8, 2004, Proceedings*, volume 3194. Springer Science & Business Media, 2004.

[7] Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[8] Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

[9] Moritz Diehl, Hans Joachim Ferreau, and Niels Haverbeke. Efficient numerical methods for nonlinear mpc and moving horizon estimation. In *Nonlinear model predictive control*, pages 391–417. Springer, 2009.

[10] David D Fan and Evangelos A Theodorou. Differential dynamic programming for time-delayed systems. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 573–579. IEEE, 2016.

[11] David D Fan, Jennifer Nguyen, Rohan Thakker, Nikhilesh Alatur, Ali-akbar Agha-mohammadi, and Evangelos A Theodorou. Bayesian learning-based adaptive control for safety critical systems. In *International Conference on Robotics and Automation*, 2020.

[12] Wagner Piazza Gaglianone, Luiz Renato Lima, Oliver Linton, and Daniel R Smith. Evaluating value-at-risk models via quantile regression. *Journal of Business & Economic Statistics*, 29(1):150–160, 2011.

[13] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[14] Yarin Gal, Rowan Thomas Mcallister, and Carl Edward Rasmussen. Improving PILCO with Bayesian Neural Network Dynamics Models. *Data-Efficient Machine Learning Workshop, ICML*, pages 1–7, 2016.

[15] Yiqi Gao, Andrew Gray, H. Eric Tseng, and Francesco Borrelli. A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles. *Vehicle System Dynamics*, 52(6):802–823, jun 2014. ISSN 0042-3114. doi: 10.1080/00423114.2014.902537.

[16] Agathe Girard, Carl Edward Rasmussen, Joaquin Quinonero Candela, and Roderick Murray-Smith. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. In *Advances in neural information processing systems*, pages 545–552, 2003.

[17] Akhil Gupta, Naman Shukla, Lavanya Marla, and Arinbjörn Kolbeinsson. Monotonic trends in deep neural networks. *arXiv preprint arXiv:1909.10662*, 2019.

[18] Lukas Hewing and Melanie N Zeilinger. Stochastic Model Predictive Control for Linear Systems using Probabilistic Reachable Sets. may 2018. doi: 10.1109/CDC.2018.8619554.

[19] Lukas Hewing, Juraj Kabzan, and Melanie N Zeilinger. Cautious Model Predictive Control using Gaussian Process Regression. *arXiv*, may 2017. URL http://arxiv.org/abs/1705.10702.

[20] Lukas Hewing, Kim P Wabersich, Marcel Menner, and Melanie N Zeilinger. Learning-Based Model Predictive Control: Toward Safe Learning in Control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 2019.

[21] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

[22] Eric C Kerrigan and Jan M Maciejowski. Robust feasibility in model predictive control: Necessary and sufficient conditions. In *Proceedings of the 40th IEEE Conference on Decision and Control*, volume 1, pages 728–733. IEEE, 2001.

[23] Roger Koenker and Gilbert Bassett Jr. Regression quantiles. *Econometrica: journal of the Econometric Society*, pages 33–50, 1978.

[24] Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15(4):143–156, 2001.

[25] Johannes Köhler, Peter Kötting, Raffaele Soloperto, Frank Allgöwer, and Matthias A Müller. A robust adaptive model predictive control framework for nonlinear uncertain systems. *arXiv preprint arXiv:1911.02899*, 2019.

[26] Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-Based Model Predictive Control for Safe Exploration. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6059–6066. IEEE, dec 2018. ISBN 978-1-5386-1395-5. doi: 10.1109/CDC.2018.8619572.

[27] Hideo Kozumi and Genya Kobayashi. Gibbs sampling methods for Bayesian quantile regression. *Journal of statistical computation and simulation*, 81(11): 1565–1578, 2011.

[28] W Langson, I Chryssochoos, S V Raković, and D Q Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125–133, jan 2004. ISSN 0005-1098. doi: 10.1016/J.AUTOMATICA.2003.08.009.

[29] Taeyoung Lee, Melvin Leok, and N Harris McClamroch. Geometric tracking control of a quadrotor uav on se (3). In *49th IEEE conference on decision and control (CDC)*, pages 5420–5425. IEEE, 2010.

[30] Anqi Liu, Guanya Shi, Soon-Jo Chung, Anima Anandkumar, and Yisong Yue. Robust Regression for Safe Exploration in Control. *arXiv preprint arXiv:1906.05819*, 2019.

[31] Brett T Lopez, Jonathan P How, and Jean-Jacques E Slotine. Dynamic tube mpc for nonlinear systems. In *2019 American Control Conference (ACC)*, pages 1655–1662. IEEE, 2019.

[32] D. Q. Mayne, E. C. Kerrigan, E. J. van Wyk, and P. Falugi. Tube-based robust nonlinear model predictive control. *International Journal of Robust and Nonlinear Control*, 21(11):1341–1353, jul 2011. ISSN 10498923. doi: 10.1002/rnc.1758.

[33] David Q Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12): 2967–2986, 2014.

[34] Duy Nguyen-Tuong, Jan Peters, Matthias Seeger, and Bernhard Schölkopf. Learning inverse dynamics: a comparison. In *European symposium on artificial neural networks*, 2008.

[35] Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot. Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4029–4036. IEEE, may 2014. ISBN 978-1-4799-3685-4. doi: 10.1109/ICRA.2014.6907444.

[36] Chris J. Ostafew, Angela P. Schoellig, and Timothy D. Barfoot. Robust Constrained Learning-based NMPC enabling reliable mobile robot path tracking. *The International Journal of Robotics Research*, 35(13):1547–1563, nov 2016. ISSN 0278-3649. doi: 10.1177/0278364916645661.

[37] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[38] Hadi Ravanbakhsh and Sriram Sankaranarayanan. Learning Control Lyapunov Functions from Counterexamples and Demonstrations. apr 2018. doi: 10.1007/s10514-018-9791-9.

[39] Filipe Rodrigues and Francisco C Pereira. Beyond expectation: Deep joint mean and quantile regression for spatio-temporal problems. *arXiv preprint arXiv:1808.08798*, 2018.

[40] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[41] Jonathan Sadeghi, Marco De Angelis, and Edoardo Patelli. Efficient training of interval Neural Networks for imprecise training data. *Neural Networks*, 118:338–351, 2019.

[42] Mattia Segú, Antonio Loquercio, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *arXiv preprint arXiv:1907.06890*, 2019.

[43] Joseph Sill. Monotonic networks. In *Advances in neural information processing systems*, pages 661–667, 1998.

[44] Sumeet Singh, Anirudha Majumdar, Jean-Jacques Slotine, and Marco Pavone. Robust online motion planning via contraction theory and convex optimization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5883–5890. IEEE, 2017.

[45] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[46] Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. In *Advances in Neural Information Processing Systems*, pages 6414–6425, 2019.

[47] James W Taylor. A quantile regression approach to estimating the distribution of multiperiod returns. *The Journal of Derivatives*, 7(1):64–78, 1999.

[48] Mario E Villanueva, Rien Quirynen, Moritz Diehl, Benoît Chachuat, and Boris Houska. Robust mpc via min–max differential inequalities. *Automatica*, 77:311–321, 2017.

[49] Kim P Wabersich and Melanie N Zeilinger. Linear model predictive safety certification for learning-based control. mar 2018. URL http://arxiv.org/abs/1803.08552.

[50] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. *IEEE Transactions on Robotics*, 34 (6):1603–1622, 2018.

[51] Xing Yan, Weizhong Zhang, Lin Ma, Wei Liu, and Qi Wu. Parsimonious quantile regression of financial asset tail dynamics via sequential learning. In *Advances in Neural Information Processing Systems*, pages 1575–1585, 2018.

[52] Yunwen Yang, Huixia Judy Wang, and Xuming He. Posterior inference in Bayesian quantile regression with asymmetric Laplace likelihood. *International Statistical Review*, 84(3):327–344, 2016.

[53] Seungil You, David Ding, Kevin Canini, Jan Pfeifer, and Maya Gupta. Deep lattice networks and partial monotonic functions. In *Advances in Neural Information Processing Systems*, pages 2981–2989, 2017.

[54] Faen Zhang, Xinyu Fan, Hui Xu, Pengcheng Zhou, Yujian He, and Junlong Liu. Regression via Arbitrary Quantile Modeling. *arXiv preprint arXiv:1911.05441*, 2019.