NONLINEAR MODEL PREDICTIVE CONTROL FOR MODELS IN QUASI-LPV FORM, P. S. GONZALEZ CISNEROS & H. WERNER, 2020

ANTJE DITTMER, DLR-FT REGELUNGSTECHNIK MEETUP, JULY 3RD 2025

Credit: DLR

Abstract

- **Nonlinear Model Predictive Control (NMPC) approach based on quasi-Linear Parameter Varying (q-LPV)** representations of the model and constraints
  - LPV: Nonlinear system represented as a collection of linear models whose dynamics vary with a set of measurable, time-varying scheduling parameters; linearisation of system at different OP
  - q-LPV: “A clear distinction between quasi-LPV models and LPV models (…) is the functional dependency of the scheduling parameter on the states (…) “
- **Stability ensured by the offline solution of an optimization problem with Linear Matrix Inequality (LMI)** constraints in conjunction with an online **terminal state constraint**.
  - LMIs: Constraints: $F(x) = F0 + x1\ F1 + \cdots + xn\ Fn \succeq 0$, Fi symmetric matrices and F(x) is positive semidefinite. Convex feasible sets that can be efficiently solved using semidefinite programming.
  - Terminal state constraint: Constraint on the last predicted state of the MPC prediction horizon
- Iterative approach: NMPC optimization is solved a series **of Quadratic Programs (QP)** at each time step for computationally efficiency

2 Problem Setup: quasi-LPV model

We consider the quasi-LPV model

$$\begin{aligned} x_{k+1} &= A(\rho_k)\, x_k + B(\rho_k)\, u_k \\ y_k &= C(\rho_k)\, x_k \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $\rho_k = f(x_k)$[1] $\in \mathbb{R}^{n_\rho}$, and $A : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n \times n}$, $B : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n \times m}$, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n_\rho}$ are continuous maps. We assume that $\rho_k \in \mathcal{P}$, $\Delta\rho_k = \rho_k - \rho_{k-1} \in \mathcal{V}\ \forall k \geq 0$ where $\mathcal{P}, \mathcal{V} \subset \mathbb{R}^{n_\rho}$ are given compact sets. This implies that A and B are bounded on $\mathcal{P}$. Throughout this paper we will also assume that $(A(\rho), B(\rho))$ is stabilizable $\forall \rho \in \mathcal{P}$. In addition, we will assume that the parameters ρ are functions of the states alone (and not the input) for ease of exposition, i.e. $f : \mathbb{R}^n \to \mathbb{R}^{n_\rho}$. A

1. Richalet J, Rault A, Testud J, Papon J. Model Predictive Heuristic Control: Application to Industrial Processes. *Automatica* 1978; 14(5): 413–428.

2 Problem Setup: Cost function and Optimization Problem

Penalty on deviation from steady-state states

$$J_k = \sum_{l=1}^{N} (\tilde{x}_{k+l}^{\mathrm{T}} Q \tilde{x}_{k+l} + \tilde{u}_{k+l-1}^{\mathrm{T}} R \tilde{u}_{k+l-1}) + \Psi(x_{k+N}) \quad (2)$$

Terminal state constraints

where $Q = Q^{\mathrm{T}} \succ 0$, $R = R^{\mathrm{T}} \succ 0$, $\Psi(x) > 0 \ \forall \ x \neq 0$ and $\Psi(0) = 0$; $\tilde{x}_{k+i} = x_{k+i} - x_{ss}$, $\tilde{u}_{k+i} = u_{k+i} - u_{ss}$ represent deviation variables from the desired steady state values x_{ss}, u_{ss} which are assumed to be known. At each sampling instant k, the values of x_k and u_{k-1} are known, and the optimization problem

Penalty on deviation from steady-state inputs

$$\begin{aligned}
&\min_{U_k} J_k(U_k) && (3a)\\
\text{subject to} \quad &x_{k+i+1} = A(\rho_{k+i})x_{k+i} + B(\rho_{k+i})u_{k+i} && (3b)\\
&u_{k+i} \in \mathcal{U}, && (3c)\\
&x_{k+i} \in \mathcal{X} \quad i = 0, 1, ..., N-1 && (3d)\\
&x_{k+N} \in \mathcal{X}_T && (3e)
\end{aligned}$$

Constrained optimization problem

is solved online for

$$U_k = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \mathbb{R}^{Nm}, \quad (4)$$

Input prediction trajectory

2 Problem Setup: Cost function cont.

$$f(\mathcal{X}) \subset \mathcal{P}, \tag{5}$$

Subset scheduling parameter set

additionally, we assume that each individual parameter is a function of a single state or input, i.e. $f_i : \mathbb{R} \to \mathbb{R}$

$$\rho_i = f_i(x_j)$$

Scheduling parameter derivable from states

and that each f_i is bijective. We can therefore map admissible parameter values into the maximum admissible state values by defining

Maximum of states from scheduling parameter

$$\bar{x}_i = \max_{\mathcal{P}} |f_j^{-1}(\rho_j)|. \tag{6}$$

This imposition on the selection of parameters might seem restrictive, it is however the natural and common choice when constructing a qLPV model, except when LFT or polytopic synthesis techniques are used, which require a redefinition of parameters, this is however not done here.

We also define

State trajectory

$$X_k = \begin{bmatrix} x_{k+1} \\ x_{k+2} \\ \vdots \\ x_{k+N} \end{bmatrix} \in \mathbb{R}^{Nn}, \qquad \mathsf{P}_k = \begin{bmatrix} \rho_k \\ \rho_{k+1} \\ \vdots \\ \rho_{k+N-1} \end{bmatrix} \in \mathbb{R}^{Nn_\rho}, \tag{7}$$

Scheduling parameter trajectory

and with a slight abuse of notation use f to denote $\mathsf{P}_k = f(\begin{bmatrix} x_k^\mathrm{T} & X_k^\mathrm{T} \end{bmatrix}^\mathrm{T})$. Note that the map f is not directly applied to X_k but to $\begin{bmatrix} x_k^\mathrm{T} & X_k^\mathrm{T} \end{bmatrix}^\mathrm{T}$, as the vector P_k contains the parameters from time k to $k+N-1$ whereas the prediction is done for time $k+1$ to $k+N$.

2 Problem Setup: Scheduling Parameter Trajectory

For a general LPV system (with exogenous parameters) it is not possible to solve the optimization problem (3), because the future state sequence cannot be predicted; indeed X_k depends not only on the future control inputs U_k, but also on the future scheduling parameters P_k, which are not assumed to be known *a priori* but only to be measurable online. In contrast, for a *quasi-LPV system*, where the scheduling parameters ρ_k are determined by x_k and u_k, the state trajectory can be predicted. Indeed, the predicted state vector $x(k+l)$ can be calculated for each $l > 0$ from $x(k)$ and $u(k+i)$, $i = 0, 1, \ldots, l-1$, as illustrated by this simple example: Assume that a single input, first order system with a single parameter $\rho_k = x_k$ is given by

$$x_{k+1} = \underbrace{(a_0 + a_1 \rho_k)}_{a(\rho_k)} x_k + b u_k,$$

substituting $\rho_k = x_k$ and $\rho_{k+1} = x_{k+1}$, an expression for $a(\rho_{k+1}) = a(x_{k+1})$ can be very easily obtained as

$$a(x_{k+1}) = a_0 + a_1 x_{k+1} = a_0 + a_1 a_0 x_k + a_1^2 x_k^2 + a_1 b u_k.$$

The value of x_{k+2} is then

$$x_{k+2} = (a_0 + a_1 a_0 x_k + a_1^2 x_k^2 + a_1 b u_k)(a_0 x_k + a_1 x_k^2 + b u_k) + b u_{k+1}$$

It is easy to see that a similar expression can be obtained for all x_{k+i}, $i = 1, 2, \ldots, N$.

3 Iterative Predictions: Scheduling Parameter Trajectory

Optimal state trajectory from initial state and optimal input trajectory

Using the definitions (4) and (7), X_k can be written in vector form as

$$X_k = \Lambda(\mathsf{P}_k)x_k + S(\mathsf{P}_k)U_k, \tag{8}$$

where

$$\Lambda(\mathsf{P}_k) = \begin{bmatrix} A(\rho_k) \\ A(\rho_{k+1})A(\rho_k) \\ \vdots \\ A(\rho_{k+N-1})A(\rho_{k+N-2})\dots A(\rho_k) \end{bmatrix} \quad U_k = \begin{bmatrix} u_k \\ u_{k+1} \\ \vdots \\ u_{k+N-1} \end{bmatrix} \in \mathbb{R}^{Nm}, \quad X_k = \begin{bmatrix} x_{k+1} \\ x_{k+2} \\ \vdots \\ x_{k+N} \end{bmatrix} \in \mathbb{R}^{Nn}$$

and

$$S(\mathsf{P}_k) = \begin{bmatrix} B(\rho_k) & 0 & \dots \\ A(\rho_{k+1})B(\rho_k) & B(\rho_{k+1}) & \dots \\ \vdots & \vdots & \dots \\ A(\rho_{k+N-1})\dots A(\rho_{k+1})B(\rho_k) & A(\rho_{k+N-1})\dots A(\rho_{k+2})B(\rho_{k+1}) & \dots \end{bmatrix}. \quad \mathsf{P}_k = \begin{bmatrix} \rho_k \\ \rho_{k+1} \\ \vdots \\ \rho_{k+N-1} \end{bmatrix} \in \mathbb{R}^{Nn_\rho}$$

3 Iterative Predictions: Cost function and Algorithm Outline

Xk - Xss

$$J_k = \left(\Lambda(\mathsf{P}_k)x_k + S(\mathsf{P}_k)U_k - X_{ss}\right)^{\mathrm{T}} \hat{Q} \left(\Lambda(\mathsf{P}_k)x_k + S(\mathsf{P}_k)U_k - X_{ss}\right) + (U_k - U_{ss})^{\mathrm{T}} \hat{R}(U_k - U_{ss}) + \Psi(x_{k+N}) \quad (9)$$

where $X_{ss} = \mathbf{1}_N \otimes x_{ss}$, $U_{ss} = \mathbf{1}_N \otimes u_{ss}$. By fixing the scheduling trajectory P_k, the predicted states X_k in (8) are linear in the control inputs U_k, and the optimization problem (3) can be solved with the same complexity as its LTI counterpart. This motivates the following iterative approach:

- A previous solution X^l_{k-1} is used to initialize P^0_k, i.e. $\mathsf{P}^0_k = f(X^l_{k-1})$ (without time-shifting X^l_k). At $k = 0$, the trajectory is assumed frozen at its initial value $\mathsf{P}^0_k = \mathbf{1}_N \otimes f(x_0)$
- The scheduling sequence P^l_k is then iteratively driven towards its optimal value $\mathsf{P}^*_k = f(X^*_k)$, where X^*_k denotes the state sequences corresponding to the optimal solution to (3).
- This is achieved by solving at iteration step l the optimization problem (3) with P_k replaced by P^l_k, and by generating a new scheduling sequence from the resulting optimal state sequence X^l_k as $\mathsf{P}^{l+1}_k = f(X^l_k)$.

When $|X^l_k - X^{l-1}_k| < \epsilon$, for some $\epsilon > 0$ the input sequence U^l_k is taken as an approximation of the optimal solution U^*_k to (3) and the first element of the sequence is applied to the plant. The proposed approach is summarized in Algorithm 1, and its convergence properties are highlighted in Figure 3 in an example, below.

3 Iterative Predictions: Algorithm

Algorithm 1 qLMPC

Require: u_{ss}, x_{ss}, plant model, $\hat{Q}$, $\hat{R}$, N

1: $k \leftarrow 0$
2: Define $\mathsf{P}^0 = \mathbf{1}_N \otimes f(x_k, u_{k-1})$ — Initial constant trajectory
3: **repeat**
4: $\quad l \leftarrow 0$
5: $\quad$ **repeat** — Find optimal control trajectory over prediction horizon
6: $\quad\quad$ Solve (3) using P_k^l for U_k^l
7: $\quad\quad$ Use (8) to calculate X_k^l using P_k^l and U_k^l — Calculate state trajectory over prediction horizon
8: $\quad\quad$ Define $\mathsf{P}_k^{l+1} = f(X_k^l, U_k^l)$ — Calculate scheduling parameter trajectory over prediction horizon
9: $\quad\quad l \leftarrow l + 1$
10: $\quad$ **until** stop criterion
11: $\quad$ Apply u_k to the system — First sample of control trajectory applied to system
12: $\quad$ Define $\mathsf{P}_{k+1}^0 = f(X_k^l, U_k^l)$ — Set scheduling parameter trajectory for next sample
13: $\quad k \leftarrow k + 1$
14: **until** end

3 Iterative Predictions: Quadratic Subproblem for QP

A QP sub-problem in SQP algorithms is generated by using a second order approximation of the cost function and a linearization of the constraints: consider the nonlinear optimization problem

$$\begin{aligned} \min \quad & V(x) \\ \text{s.t.} \quad & h(x) = 0 \\ & g(x) < 0 \end{aligned}$$

a quadratic sub-problem is given by

Description

Solver for quadratic objective functions with linear constraints.

quadprog finds a minimum for a problem specified by

$$\min_x \frac{1}{2} x^T H x + f^T x \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

H, A, and Aeq are matrices, and f, b, beq, lb, ub, and x are vectors.

0.5

$$\min \quad \tilde{x}^{\mathrm{T}} HV(x^l)\tilde{x} + \nabla V(x^l)^{\mathrm{T}}\tilde{x} \tag{13}$$

$$\text{s.t.} \quad \nabla h(x^l)\tilde{x} + h(x^l) = 0 \tag{14}$$

$$\nabla g(x^l)\tilde{x} + g(x^l) < 0 \tag{15}$$

where $HV(x^l)$ is the Hessian of the cost function, x^l is the current solution estimate (at iteration l), and $\tilde{x} = x - x^l$.

4 State Constraints

The cost function (9) does not include the state sequence as variables. The predicted state was eliminated from the cost function by making the substitution $X_k = \Lambda(\mathsf{P}_k)x_k + S(\mathsf{P}_k)U_k$. Using this same substitution, state constraints can be included in the optimization problem, these will be converted into additional constraints on the input trajectory U_k.

We next define the *constrained output* $y_c(k) = h(x_k) = C_c(\rho_k)x_k$; note that this is in general different from the tracking output in (1), although it might coincide in a given application. The predicted constrained output is thus

$$Y_c(k) = \bar{C}_c(\mathsf{P}_k^c)X_k = \bar{C}_c(\mathsf{P}_k^c)(\Lambda(\mathsf{P}_k)x_k + S(\mathsf{P}_k)U_k)$$

where $\bar{C}_c(\mathsf{P}_k^c) = \mathrm{diag}(C_c(\rho_{k+1}), C_c(\rho_{k+2}), ..., C_c(\rho_{k+N}))$, and the parameter vector $\mathsf{P}_k^c = f_c(X_k) = \begin{bmatrix}\rho_{k+1}^c & \rho_{k+2}^c & \cdots & \rho_{k_N}^c\end{bmatrix}$ is used to schedule the constraint. Note that this may differ from the scheduling vector used for the plant (1). Given a constraint $y_c(k+i) \leq b_c$ $\forall i = 1, ..., N$ where b_c is constant, we can use the definitions above to constrain the predicted state and by extension the input trajectory as

$$C_c(\mathsf{P}_k^c)S(\mathsf{P}_k)U_k \leq \mathbf{1}b_c - C_c(\mathsf{P}_k^c)\Lambda(\mathsf{P}_k)x_k.$$

Using this formulation, nonlinear constraints can be accommodated by finding a quasi-LPV representation of $h(x_k)$ and making C_c parameter dependent. By using the predicted parameter trajectory P_k^c this constraint becomes linear and is written in the standard form $Ax \leq b$, this turns problem (3) into a QP.

5 Stability Analysis: Stabilizing MPC for qLPV

Theorem 1 (Stabilizing MPC for quasi-LPV systems)**.** Assume a nominal control law $u_k^{\text{nom}} = F(\rho_k)x_k$, a terminal state domain $\mathcal{X}_T(\rho_k)$ and a terminal cost $\Psi(x_k, \rho_k)$ exist such that $\forall \rho \in \mathcal{P}$ the following conditions hold.

1. $0 \in \text{int}\left(\mathcal{X}_T(\rho)\right)$ — origin lies in interior of terminal set
2. $(A(\rho) + B(\rho)F(\rho))x \in \mathcal{X}_T(\rho), \qquad \forall x \in \mathcal{X}_T(\rho)$ — Once a state is in terminal set, all consecutive states are
3. $\Psi((A(\rho_k) + B(\rho_k)F(\rho_k))x_k, \rho_{k+1}) - \Psi(x_k, \rho_k) \leq -x_k^{\mathrm{T}} Q x_k - x_k^{\mathrm{T}} F(\rho_k)^{\mathrm{T}} R F(\rho_k) x_k, \qquad \forall x_k \in \mathcal{X}_T(\rho_k)$ — Lyapunov terminal cost decrease condition
4. $F(\rho)x \in \mathcal{U}, \qquad \forall x \in \mathcal{X}_T(\rho)$ — control input from state in terminal set constraints
5. $\mathcal{X}_T(\rho) \subset \text{Im}(f^{-1}(\mathcal{P}))$ — terminal set lies with inverse scheduling map
6. $\mathcal{X}_T(\rho) \subset \mathcal{X}$ — terminal set lies within the state constraint set

Then, assuming feasibility of the initial state, an MPC controller solving the optimization problem (3) guarantees asymptotic stability of the origin.

5 Stability Analysis: Stabilizing MPC for qLPV cont.

2. $(A(\rho) + B(\rho)F(\rho))x \in \mathcal{X}_T(\rho), \qquad \forall x \in \mathcal{X}_T(\rho)$
3. $\Psi((A(\rho_k) + B(\rho_k)F(\rho_k))x_k, \rho_{k+1}) - \Psi(x_k, \rho_k) \leq -x_k^{\mathrm{T}} Q x_k - x_k^{\mathrm{T}} F(\rho_k)^{\mathrm{T}} R F(\rho_k) x_k, \qquad \forall x_k \in \mathcal{X}_T(\rho_k)$
4. $F(\rho)x \in \mathcal{U}, \qquad \forall x \in \mathcal{X}_T(\rho)$
5. $\mathcal{X}_T(\rho) \subset \mathrm{Im}(f^{-1}(\mathcal{P}))$
6. $\mathcal{X}_T(\rho) \subset \mathcal{X}$

Then, assuming feasibility of the initial state, an MPC controller solving the optimization problem (3) guarantees asymptotic stability of the origin.

Proof. Follows the same lines as Theorem 1 in [16]. □

Terminal cost

Terminal set

The conditions in Theorem 1 are enforced by the offline solution of an LMI problem, the constraints of which can be imposed on a grid over the admissible parameter set. We discuss now how to choose $\Psi(x)$ and $\mathcal{X}_T(x)$ to arrive at a convex optimization problem. Consider the quadratic terminal cost $\Psi(x) = x^{\mathrm{T}} P(\rho) x$ and the sublevel set of the Lyapunov function given by

$$\mathcal{X}_T = \{x | x^{\mathrm{T}} P(\rho) x \leq \alpha\} \tag{16}$$

with $\alpha > 0$. To guarantee that this set is invariant under the control law $u = F(\rho)x$, we need to make sure that $F(\rho)x \in \mathcal{U}$ $\forall x \in \mathcal{X}$. Furthermore, we would want this set to be as large as possible, therefore we want to maximize the value of α, subject to the input constraints. For this we make use of the following lemma.

5 Stability Analysis: LMI

Theorem 2. Let $Y(\rho) = P(\rho)^{-1}$, $X(\rho) = F(\rho)Y(\rho)$ and $\tilde{\alpha} = 1/\alpha$, then $\Psi(x, \rho) = x^{\mathrm{T}} P(\rho)x$, $\mathcal{X}_T(\rho) = \{x | x^{\mathrm{T}} P(\rho)x \leq \alpha\}$ and $F(\rho)$ satisfy the conditions of Theorem 1 if $Y(\rho)$ and $X(\rho)$ and $\tilde{\alpha}$ are chosen as the solutions to the following optimization problem

$$\min_{X,Y,\tilde{\alpha}} \ \tilde{\alpha} \tag{17a}$$

s.t.

$$\begin{bmatrix} Y(\rho) & * & * & * \\ (A(\rho)Y(\rho) + B(\rho)X(\rho)) & Y(\rho + \Delta\rho) & * & * \\ Y(\rho) & 0 & Q^{-1} & * \\ X(\rho) & 0 & 0 & R^{-1} \end{bmatrix} \succeq 0, \tag{17b}$$

$$\begin{bmatrix} \tilde{\alpha} u_{i,max}^2 & e_i X(\rho) \\ * & Y(\rho) \end{bmatrix} \succeq 0 \qquad i \in [1\ m] \tag{17c}$$

$$e_j Y(\rho) e_j^{\mathrm{T}} \leq \tilde{\alpha} \overline{x}_{j,max}^2 \qquad j \in [1\ n_\rho] \tag{17d}$$

$$Y(\rho) \succ 0 \quad \tilde{\alpha} > 0 \qquad \forall \rho \in \mathcal{P},\ \Delta\rho \in \mathcal{V} \tag{17e}$$

In the above inequalities, $u_{max,i}$ denotes an upper bound on $|u_i|$, e_j is the j^{th} row of the identity matrix and $\overline{x}_j$ is the upper bound on parameter state x_j as given by (6).

5 Stability Analysis: Schur Complement (Condition 3)

Proof. Satisfaction of condition 1 follows immediately from the definition of the ellipsoids (16).

(17b) $\Rightarrow$ *condition 3:*

Apply Schur complement to (17b) to get

$$\begin{bmatrix} Q^{-1} & 0 \\ 0 & R^{-1} \end{bmatrix} \succeq 0$$

$$\begin{bmatrix} Y(\rho) - Y(\rho)QY(\rho) - Y(\rho)F(\rho)^{\mathrm{T}}RF(\rho)Y(\rho) & Y(\rho)(A(\rho) + B(\rho)F(\rho))^{\mathrm{T}} \\ (A(\rho) + B(\rho)F(\rho))Y(\rho) & Y(\rho + \Delta\rho) \end{bmatrix} \succeq 0$$

The first condition is trivially fulfilled. Another use of the Schur complement to the second condition yields

$$Y(\rho + \Delta\rho) \succeq 0$$

$$Y(\rho)(A(\rho) + B(\rho)F(\rho))^{\mathrm{T}}Y(\rho + \Delta\rho)^{-1}(A(\rho) + B(\rho)F(\rho))Y(\rho) - Y(\rho) \preceq -Y(\rho)QY(\rho) - Y(\rho)F(\rho)^{\mathrm{T}}RF(\rho)Y(\rho).$$

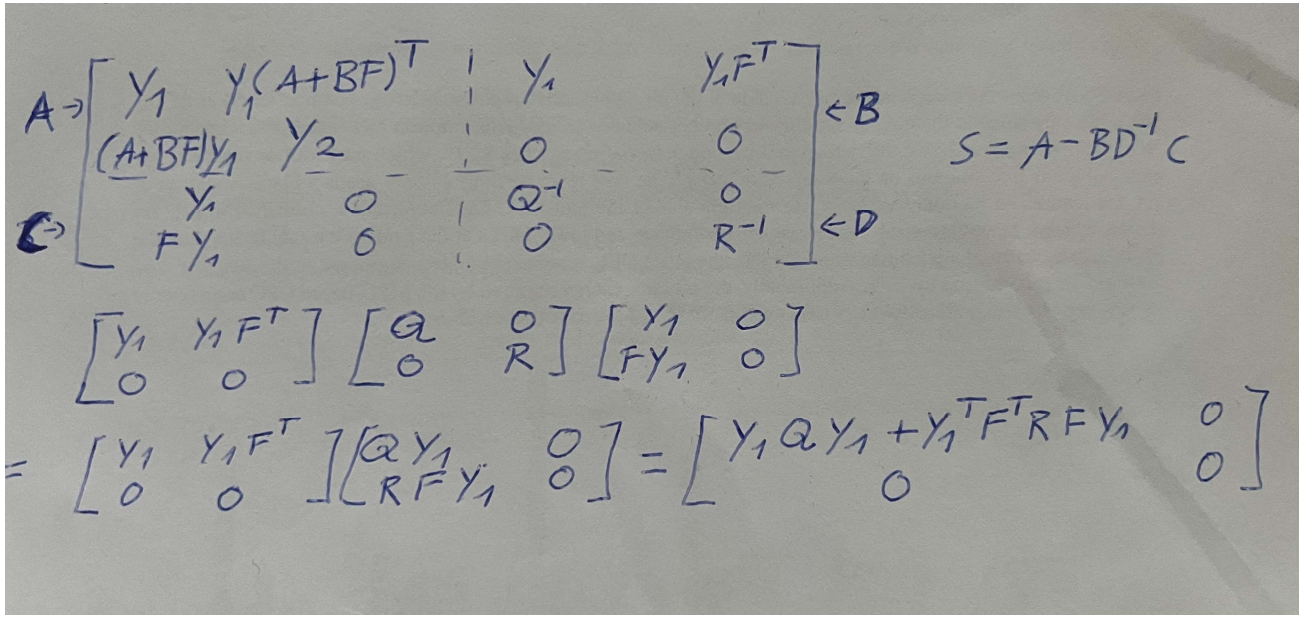
We proceed by pre- and post-multiplying the second condition by $P(\rho) = Y(\rho)^{-1}$, and by x^{T} and x,

$$x^{\mathrm{T}}(A(\rho) + B(\rho)F(\rho))^{\mathrm{T}}P(\rho + \Delta\rho)(A(\rho) + B(\rho)F(\rho))x_k - x^{\mathrm{T}}P(\rho)x \leq -x^{\mathrm{T}}Qx - x^{\mathrm{T}}F(\rho)^{\mathrm{T}}RF(\rho)x.$$

which is exactly condition 3.

5 Stability Analysis: Schur Complement Calculation 1

$$\begin{bmatrix} Y(\rho) & * & * & * \\ (A(\rho)Y(\rho)+B(\rho)X(\rho)) & Y(\rho+\Delta\rho) & * & * \\ Y(\rho) & 0 & Q^{-1} & * \\ X(\rho) & 0 & 0 & R^{-1} \end{bmatrix} \succeq 0 \Rightarrow \begin{bmatrix} Y(\rho)-Y(\rho)QY(\rho)-Y(\rho)F(\rho)^{\mathrm{T}}RF(\rho)Y(\rho) & Y(\rho)(A(\rho)+B(\rho)F(\rho))^{\mathrm{T}} \\ (A(\rho)+B(\rho)F(\rho))Y(\rho) & Y(\rho+\Delta\rho) \end{bmatrix} \succeq 0$$

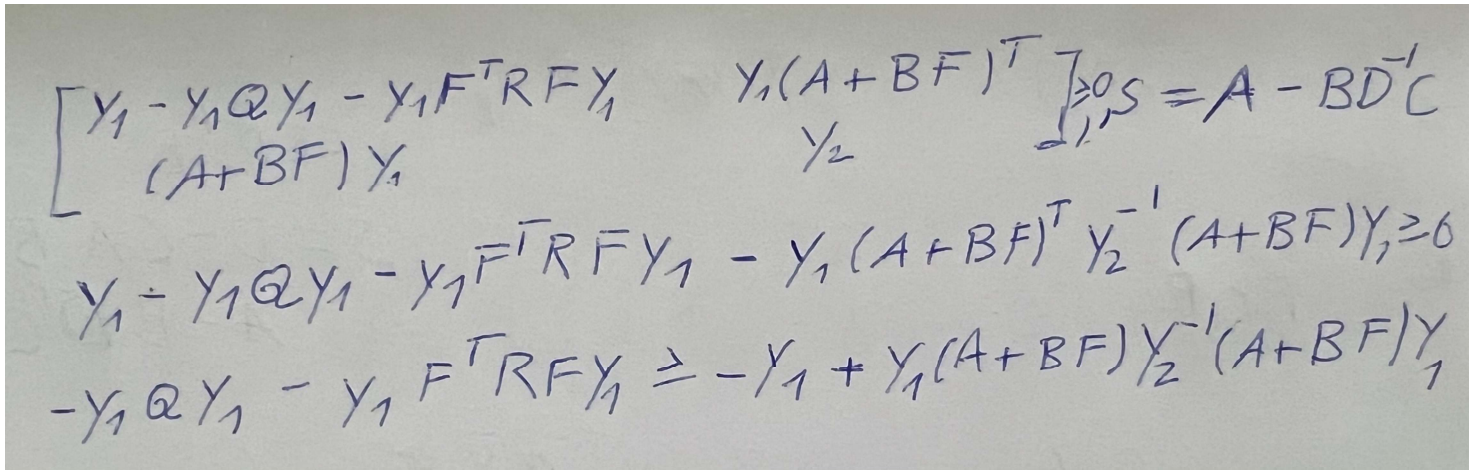


5 Stability Analysis: Schur Complement Calculation 2

$$\begin{bmatrix} Y(\rho) - Y(\rho)QY(\rho) - Y(\rho)F(\rho)^{\mathrm{T}}RF(\rho)Y(\rho) & Y(\rho)(A(\rho)+B(\rho)F(\rho))^{\mathrm{T}} \\ (A(\rho)+B(\rho)F(\rho))Y(\rho) & Y(\rho+\Delta\rho) \end{bmatrix} \geq 0$$

$$Y(\rho+\Delta\rho) \geq 0$$

$$Y(\rho)(A(\rho)+B(\rho)F(\rho))^{\mathrm{T}}Y(\rho+\Delta\rho)^{-1}(A(\rho)+B(\rho)F(\rho))Y(\rho) - Y(\rho) \leq -Y(\rho)QY(\rho) - Y(\rho)F(\rho)^{\mathrm{T}}RF(\rho)Y(\rho).$$



$$x^{\mathrm{T}}(A(\rho)+B(\rho)F(\rho))^{\mathrm{T}}P(\rho+\Delta\rho)(A(\rho)+B(\rho)F(\rho))x_k - x^{\mathrm{T}}P(\rho)x \leq -x^{\mathrm{T}}Qx - x^{\mathrm{T}}F(\rho)^{\mathrm{T}}RF(\rho)x.$$

3. $$\Psi((A(\rho_k)+B(\rho_k)F(\rho_k))x_k, \rho_{k+1}) - \Psi(x_k,\rho_k) \leq -x_k^{\mathrm{T}}Qx_k - x_k^{\mathrm{T}}F(\rho_k)^{\mathrm{T}}RF(\rho_k)x_k, \qquad \forall x_k \in \mathcal{X}_T(\rho_k)$$

5 Stability Analysis: Schur Complement (Condition 4)

(17c) ⇒ *condition 4:*

Apply Schur complement to (17c) and substitute $FY = X$

$$\tilde{\alpha} u_{i,max}^2 - e_i F Y F^{\mathrm{T}} e_i^{\mathrm{T}} \geq 0$$

$$\begin{bmatrix} \tilde{\alpha} u_{i,max}^2 & e_i X(\rho) \\ * & Y(\rho) \end{bmatrix} \geq 0 \qquad i \in [1\ m]$$

rearrange to obtain

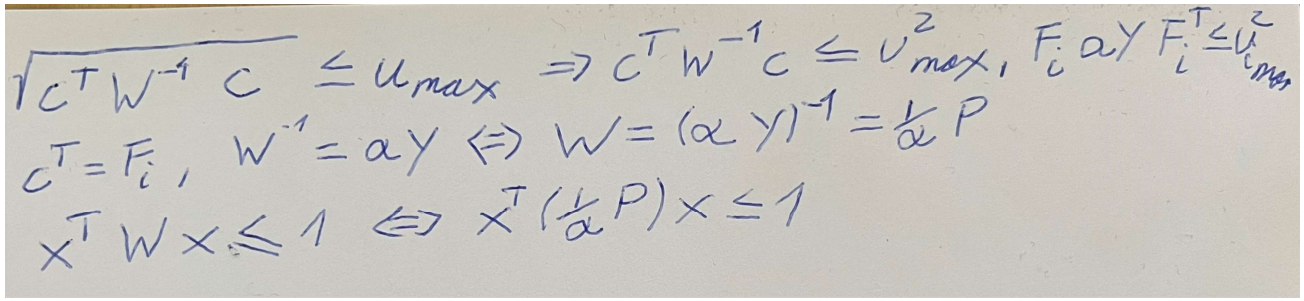
$$F_i(\alpha Y) F_i^{\mathrm{T}} \leq u_{i,max}^2 \qquad \tilde{\alpha} = 1/\alpha$$

where F_i is the i^{th} row of F. Using Lemma 1*, this gives a condition for the i^{th} input to be upper bounded by $|u_{i,max}|$ within the ellipsoidal set

$$x^{\mathrm{T}} \left(\frac{1}{\alpha} P \right) x \leq 1 \qquad Y(\rho) = P(\rho)^{-1}$$

or equivalently $x^{\mathrm{T}} P x \leq \alpha$. Invariance of this set (condition 2) follows from condition 3 (given that it is a sublevel set), proved above.

Lemma 1* (Lemma 4.1 from Löfberg[20]). The maximum of $c^{\mathrm{T}} x$ in the ellipsoidal set $x^{\mathrm{T}} W x \leq 1$ is $\sqrt{c^{\mathrm{T}} W^{-1} c}$



5 Stability Analysis: Remarks

Remark 1. Solving the LMI problem for all admissible parameters values is conservative since the conditions need only to hold within the terminal region $\mathcal{X}$. However, the terminal region is calculated only after the solutions of the LMI problem. Likewise, further conservatism is introduced by replacing $P(\rho(k+1))$ by $P(\rho+\Delta\rho)$and considering ρ and $\Delta\rho$ as two independent variables.

Remark 2. Compared with the BMI formulation in Cisneros and Werner,[18] this approach requires the solution of an LMI problem, which is easily solvable, so that no sequential or iterative procedure for the solution is necessary. On the other hand, this comes at the cost of requiring the terminal set to fulfill a stronger condition (17), rather than the milder condition (8) from Cisneros and Werner[18]. This could potentially lead to conservatism in the sense that the optimal solution to the BMI formulation may be better than the one obtained with the presented approach.

Remark 3. Whereas the controller $F(\rho)$ is a fictitious construct to prove stability, both P and $W = 1/\alpha P$ impact the feasibility and performance of the control law. In particular, it may happen that a large P is found, coupled with a large α, so that the ellipsoid (characterized by P/α) is large, however a very large value of P is not desired since it may lead to a loss in performance. In order to keep the value of P within an acceptable range, the optimization objective (17a) may be modified, e.g. to be $\min_{X,Y,\tilde{\alpha},t} \tilde{\alpha} - at$ with the additional LMI $Y(\rho) > t$, $\forall \rho \in \mathcal{P}$, where $a > 0$ is a tuning parameter to weight the desire to make the ellipsoid large or P small.

6 Tracking

- Cost function (2) penalizes deviations from the steady state values of inputs and states
- Do not provide a reference to the state, but rather a subset (or linear combination) of the states
- Information on steady state values for all states and inputs, make use of a target selector (..) to find steady state values for states and inputs in the presence of constraints

The target values are the solutions of the optimization problem

Matrix C as selector of states to be compared to steady state

$$\min_{x_s,u_s} |u_s - u_{sp}|^2_{R_s} + |C(\rho_s)x_s - y_{sp}|^2_{Q_s} \tag{18}$$

s.t.

$$\begin{bmatrix} I - A(\rho_s) & -B(\rho_s) \\ C(\rho_s) & 0 \end{bmatrix} \begin{bmatrix} x_s \\ u_s \end{bmatrix} = \begin{bmatrix} 0 \\ y_{sp} \end{bmatrix}$$

xs = Axs + Bus ⇔ (I-A)xs –Bus = 0

$$u_s \in \mathcal{U}, \qquad x_s \in X_{cons}$$

where u_{sp} and y_{sp} are setpoint values for the input and tracking output, respectively and $\rho_s = f(x_s, u_s)$.

6 Tracking: Number of tracking outputs

It is important to note that in order for a solution to exist it must hold that $l \leq m$, that is, the number of tracking outputs must be less than or at most equal to the number of inputs. Integral action can be added by state augmentation either by considering input increments, Δu or by adding an integrating disturbance model. Both these methods are standard and well known and are not discussed here[23,21].

6 Tracking: Terminal Constraint Set and Cost

One crucial issue is that the stability conditions need to be modified. The presented stability conditions rely on a terminal constraint set and a terminal cost that makes the cost function a Lyapunov function. For the tracking case appropriate modifications need to be made to keep the stability guarantee. The most straightforward, yet very conservative, choice is to impose a terminal equality constraint $x_N = x_s$, i.e. reduce the terminal set to a point. This constraint would likely need a very long prediction horizon to make the problem feasible. Other methods proposed in the literature involve partitioning the state space and computing, for each partition, terminal ingredients[24];such an approach is compatible with the presented framework, however it is not practical when the state space is high dimensional and therefore will not be pursued here. A similar, yet more practical approach to stabilizing MPC in tracking scenarios, is to leverage the input-output quasi-LPV framework such that partitions are only necessary in the output space (which is lower dimensional). This issue has been studied and reported separately.[25]

To construct a suitable terminal cost, a pragmatic choice is to take the one calculated for the regulator problem: define $\Psi(x_N) = x_N^T P x_N$ where P and F are the solutions to the feasibility LMI (17b). Parameter dependency on P and F can be dropped if the system is quadratically stabilizable, i.e. if there exists a constant P such that the above condition is met $\forall \rho \in \mathcal{P}$ otherwise a parameter dependent Lyapunov function and state feedback gain can be used.

7.1 Example 1: Stabilizing qLMPC: CSTR Parameters

Continuous stirred tank reactor (CSTR)

$$\dot{C}_A = \frac{q}{V}(C_{Af} - C_A) - k_o e^{\left(\frac{-E}{RT}\right)} C_A \tag{19}$$

$$\dot{T} = \frac{q}{V}(T_f - T) - \frac{\Delta H}{\rho C_\rho} k_o e^{\left(\frac{-E}{RT}\right)} C_A + \frac{UA}{V\rho C_\rho}(T_c - T)$$

where C_A is the concentration of A in the reactor, T is the temperature in the reactor and T_c is the coolant temperature. Pa

The objective is to drive C_A and T to the desired value $\bar{x} = [0.5 \quad 350]$ $0 \leq C_A \leq 1$ mol/l, 280 K $\leq T \leq$ 370 K, 280 K $\leq T_c \leq$ 370 K.

Parameter	Description	Value
q	Input flow	100 l/min
V	Reactor volume	100 l
C_{Af}	Feed concentration	1mol/l
k_o	Pre-exponential factor	7.2×10^{10} 1/min
E/R	Activation energy/Gas constant	8750 K
T_f	Feed temperature	350 K
ΔH	Heat of reaction	-5×10^4 J/mol
p	Density of A-B mixture	1000 g/l
C_ρ	Specific heat	0.239 J/g K
UA	Heat transfer	5×10^4 J/min K

7.1 Example 1: Stabilizing qLMPC: Jacobian

Non-linearities

Continuous stirred tank reactor (CSTR)

$$\dot{C}_A = \frac{q}{V}(C_{Af} - C_A) - k_o e^{\left(\frac{-E}{RT}\right)} C_A \qquad \text{Control input} \qquad (19)$$

$$\dot{T} = \frac{q}{V}(T_f - T) - \frac{\Delta H}{\rho C_\rho} k_o e^{\left(\frac{-E}{RT}\right)} C_A + \frac{UA}{V\rho C_\rho}(T_c - T)$$

where C_A is the concentration of A in the reactor, T is the temperature in the reactor and T_c is the coolant temperature. Parameter

$$\dot{\tilde{x}} \approx \nabla_x f(x,u)|_\rho \tilde{x} + \nabla_u f(x,u)|_\rho \tilde{u} \qquad \rho = [C_A \quad T]^{\mathrm{T}} \qquad (20)$$

where

$$\nabla_x f|_\rho = \begin{bmatrix} -\frac{q}{V} - k_o e^{\left(\frac{-E}{R\rho_2}\right)} & -\frac{Ek_o}{R\rho_2^2} e^{\left(\frac{-E}{R\rho_2}\right)} \rho_1 \\ -\frac{\Delta H k_o}{pC_\rho} e^{\left(\frac{-E}{R\rho_2}\right)} & -\frac{q}{V} - \frac{UA}{VpC_\rho} - \frac{E\Delta H k_o}{R\rho_2^2 pC_\rho} e^{\left(\frac{-E}{R\rho_2}\right)} \rho_1 \end{bmatrix}$$

$$\nabla_u f|_\rho = \begin{bmatrix} 0 \\ \frac{UA}{VpC_\rho} \end{bmatrix}$$

7.1 Example 1: Velocity-based form

$$\nabla_x f|_\rho = \begin{bmatrix} -\frac{q}{V} - k_o e^{\left(\frac{-E}{R\rho_2}\right)} & -\frac{Ek_o}{R\rho_2^2} e^{\left(\frac{-E}{R\rho_2}\right)} \rho_1 \\ -\frac{\Delta H k_o}{pC_\rho} e^{\left(\frac{-E}{R\rho_2}\right)} & -\frac{q}{V} - \frac{UA}{VpC_\rho} - \frac{E\Delta H k_o}{R\rho_2^2 pC_\rho} e^{\left(\frac{-E}{R\rho_2}\right)} \rho_1 \end{bmatrix} \qquad \rho = [C_A \quad T]^{\mathrm{T}}$$

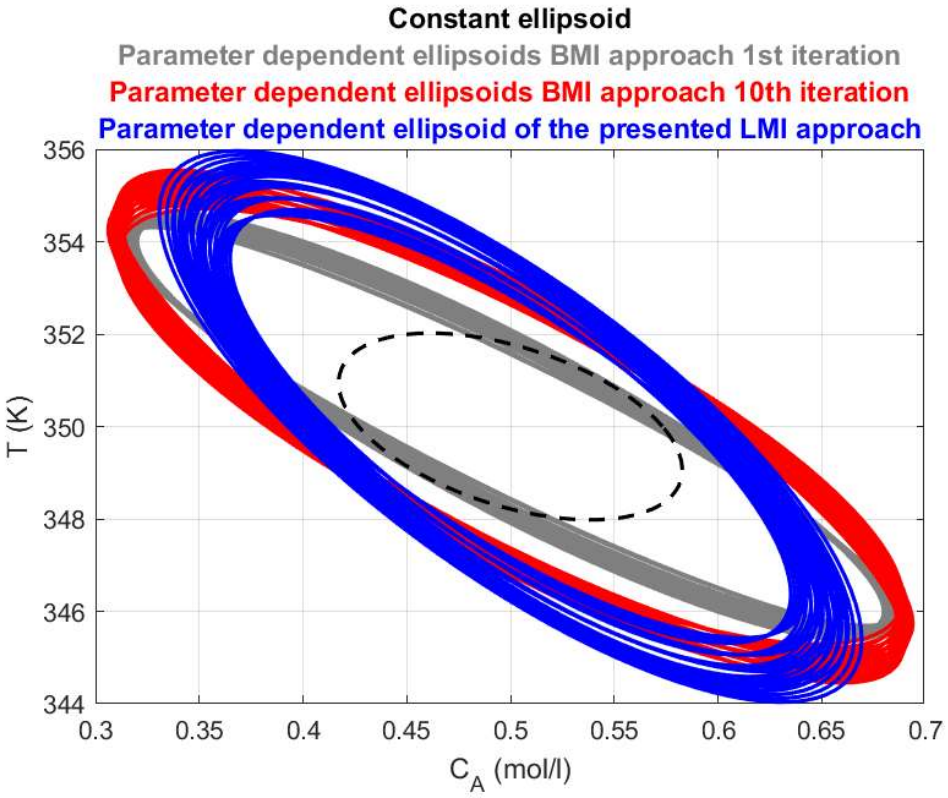
$$\nabla_u f|_\rho = \begin{bmatrix} 0 \\ \frac{UA}{VpC_\rho} \end{bmatrix}$$

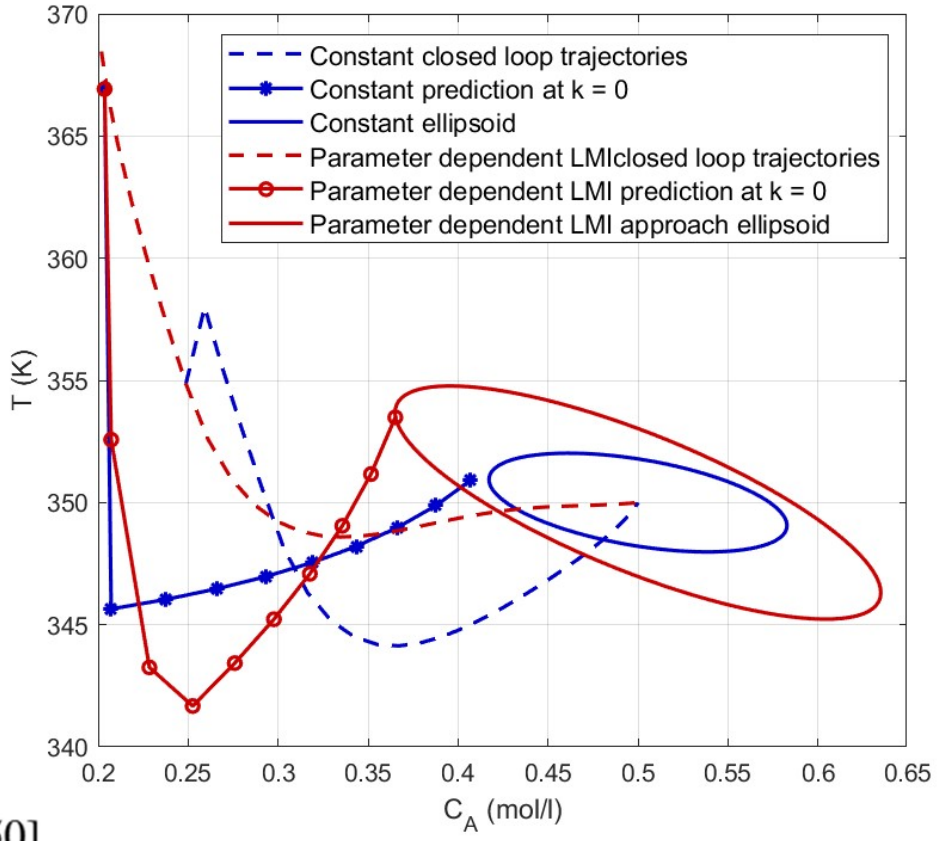
$$\ddot{x} = \nabla_x f(x,u)|_p \dot{x} + \nabla_u f(x,u)|_p \dot{u}. \tag{21}$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & I \\ 0 & \nabla_x f(\rho) \end{bmatrix}}_{A_c} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \nabla_u f(\rho) \end{bmatrix}}_{B_c} \dot{u} \tag{22}$$

An Euler discretization is done for both the states and the input (leading to a redefinition of the input as input increments) by substituting $A = (I + T_s A_c)$ and $B = B_c$[3]. For this implementation the tuning matrix Q is augmented with a block-zero matrix to avoid penalization of the velocity states to be coherent with the tuning matrices of the offline computations. A prediction horizon of $N = 10$ was used for the simulations.

7.1 Example 1: Stabilizing qLMPC: Figures





The objective is to drive C_A and T to the desired value $\bar{x} = [0.5 \quad 350]$

$$0 \leq C_A \leq 1 \text{ mol/l}, 280 \text{ K} \leq T \leq 370 \text{ K}, 280 \text{ K} \leq T_c \leq 370 \text{ K}.$$

LMI optimization problem be feasible from $k = 0$ → guaranteeing stability

7.2 Example 2: Fast implementation of qLMPC: Unicycle

- Unicycle model: minimizing online computation complexity is a priority, relinquish terminal constraint (and stability guarantees) → less expensive QP

$$\dot{x} = v\cos(\theta) \quad \dot{v} = \frac{1}{m}F \quad \dot{\omega} = \frac{1}{I}\tau$$
$$\dot{y} = v\sin(\theta) \quad \dot{\theta} = \omega$$

As in the previous example, a velocity-based linearization is carried out, keeping the positions (i.e. integrated velocities) in the state vector to be able to constrain them. A state space model is thus given by (22) (in this case one state, ω, is repeated so we can ignore one occurrence) where

$$\nabla_x f|_\rho = \begin{bmatrix} 0 & 0 & \cos(\rho_1) & -\rho_2\sin(\rho_1) & 0 \\ 0 & 0 & \sin(\rho_1) & \rho_2\cos(\rho_1) & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \nabla_u f|_\rho = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1/m & 0 \\ 0 & 0 \\ 0 & 1/I \end{bmatrix}, \qquad \rho = [\theta \quad v].$$

x y v theta omega F tau

7.2 Example 2: Fast implementation of qLMPC: Params

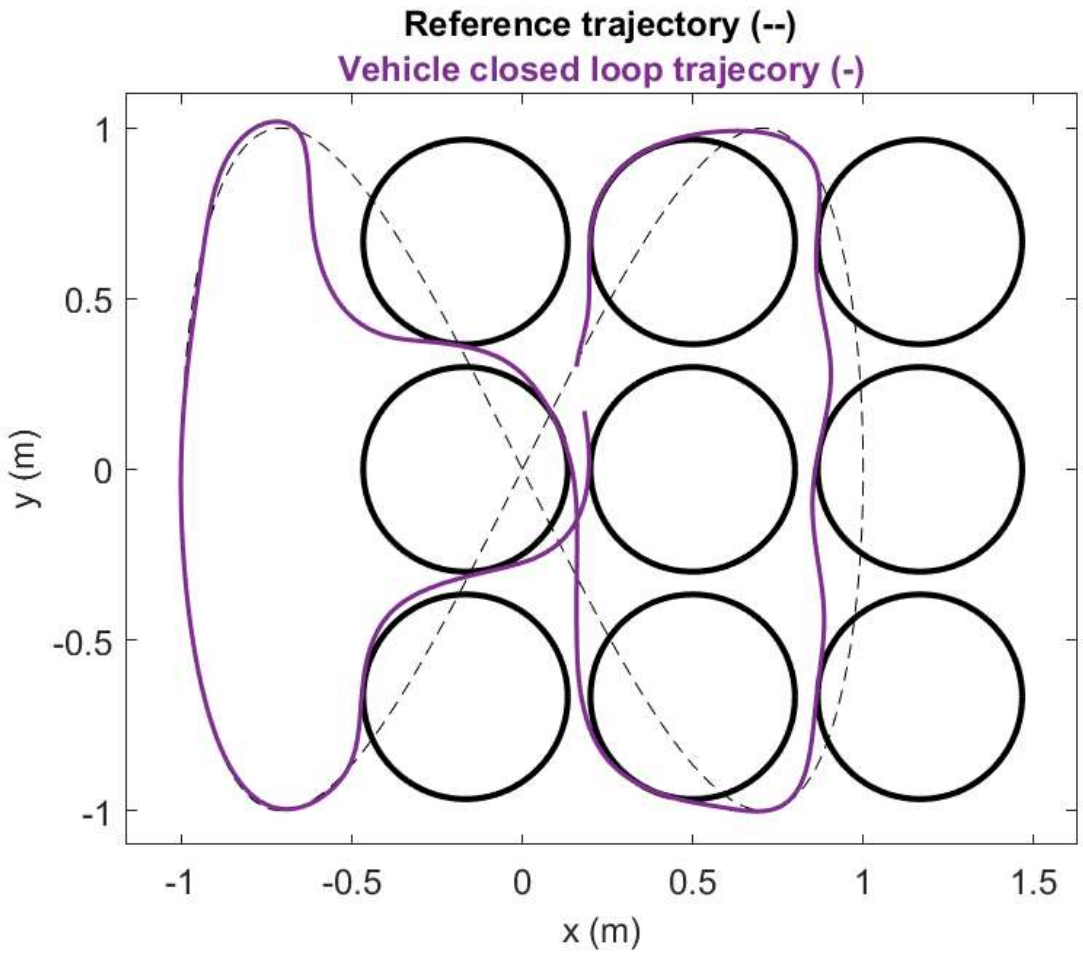
- Unicycle model: minimizing online computation complexity is a priority, relinquish terminal constraint (and stability guarantees) → less expensive QP

TABLE 2 Parameters and reference trajectory

Param.	Description	Value
m	Vehicle mass	1 kg
I	Vehicle rot. inertia	0.015 kgm^2
r_o	Obstacles' radii	0.3 m
(c_{ox}, c_{oy})	Obstacles' centers	$\{(-1/6, -2/3), (-1/6, 0), (-1/6, 2/3),$ $\{(1/2, -2/3), (1/2, 0), (1/2, 2/3),$ $\{(7/6, -2/3), (7/6, 0), (7/6, 2/3)\}$ m
x_{ref}	X reference	$\sin(1/3t)$ m
y_{ref}	Y reference	$\sin(2/3t)$ m

$$Q = \text{diag}(100, 100, 0.01, 0.1, 0.01, 0.01, 1, 0.01, 0.01) \qquad R = \text{diag}(0.01, 0.1)$$

7.2 Example 2: Figure



- Controller full knowledge of trajectory
- Reference trajectory not feasible
- Sampling time Ts = 25s, convergence time 2 iterations: Tc = 3.6

7.3 Discussion

- CSTR example:
  - Reduced conservatism of parameter dependent stability conditions
  - Not addressed: stabilizing MPC for changing set points
    - in this paper: shifting the origin and linearizing around set point
    - In other studies: partition of state space and terminal ingredients computed for each partition → not practical
- Unicycle example:
  - focused on practicality/computation speed, no stability guarantees
  - Reason for no terminal state constraints: Far away set points would need a long (prediction) horizon, although dynamically no difference from close set points
  - Future research: terminal constraints only on velocity vector → larger feasible set

8 CONCLUSIONS

- Presented algorithm: Practical approaches with and without stability guarantees
  - Stability guarantees also geared towards practical implementation
- quasi-LPV model: Original nonlinear system → scheduled linear system for efficient optimization
  - scheduling variables depend on state: Can be predicted based over prediction horizon based on initial state
  - Stability is enforced by offline solution to an LMI problem
  - parameter dependency can potentially lead to a larger terminal set
- Demonstrated on two examples

Thank you very much for your attention!

Model Predictive Control for Wind Farms

- Objectives for wind farm MPC: Minimize **error**, **change in turbine controls**, and **yaw angle**

$$\min_{\boldsymbol{U}_k} J(\boldsymbol{U}_k) = \boldsymbol{E}_k^T \boldsymbol{Q} \boldsymbol{E}_k + \Delta \boldsymbol{U}_k^T \boldsymbol{R} \Delta \boldsymbol{U}_k + \boldsymbol{U}_k^T \boldsymbol{R}_\gamma \boldsymbol{U}_k$$

$$s.t.$$

$$\begin{bmatrix} \boldsymbol{\zeta}_{d,k+1} \\ P_{WF,k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{\hat{\boldsymbol{K}}} & \boldsymbol{B}_{\hat{\boldsymbol{K}}} \\ \boldsymbol{C}_{\hat{\boldsymbol{K}}} & \boldsymbol{D}_{\hat{\boldsymbol{K}}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\zeta}_{d,k} \\ \boldsymbol{u}_k \end{bmatrix}, \quad k \in \{1, 2, \ldots, n_h - 1\}$$

$$e = P_{ref} - P_T = [e_1, e_2, \ldots, e_{n_h}]^T \in \mathbb{R}^{n_h}$$

$$\Delta u = \left[u_1^T - c^T, u_2^T - u_1^T, \ldots, u_{n_h}^T - u_{n_h-1}^T\right]^T \in \mathbb{R}^{n_h n_u}$$

- Use **Koopman wind farm models** to predict wind farm states over a **finite time horizon**
- Solve **optimization problem** to obtain **optimal control inputs** at **minimal yaw angle** for each turbine under input constraints
- Set the next control inputs and update predictions and optimisation problem at each time step