Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 51-18 (2018) 399–404

Fast Sensitivity-Based Nonlinear Economic Model Predictive Control with Degenerate NLP

Eka Suwartadi* Johannes Jäschke*

* *Department of Chemical Engineering, Norwegian University of Science and Technology (NTNU), 7491 Trondheim, Norway (e-mail: eka.suwartadi@ntnu.no and jaschke@ntnu.no).*

Abstract: We present a fast sensitivity-based nonlinear model predictive control (NMPC) algorithm, that can handle non-unique multipliers in the discretized dynamic optimization problem. Non-unique multipliers may arise, for example when path constraints are active for longer periods of the prediction horizon. This is a common situation in economic model predictive control. In such cases, the optimal nonlinear programming (NLP) solution often satisfies the Mangasarian-Fromovitz constraint qualification (MFCQ), which implies non-unique, but bounded multipliers. Consequently, any sensitivity-based fast NMPC scheme must allow for discontinuous jumps in the multipliers. In this paper, we apply a sensitivity-based path-following algorithm that allows multiplier jumps within the advance-step NMPC (asNMPC) framework. The path-following method consists of a corrector and a predictor step, which are computed by solving a system of linear equations, and a quadratic programming problem, respectively, and a multiplier jump step determined by the solution of a linear program. We demonstrate the proposed method on an economic NMPC case study with a CSTR.

Keywords: Optimal control, Economic model predictive control, Parametric optimization, Path-following.

1. INTRODUCTION

A common practice in the process industries is to divide the control system into two layers: an economic real-time optimization (RTO) layer and a lower level model predictive control (MPC) layer. Since the RTO is typically performed using a steady-state model, the two-layer control system is not able to handle transient optimally. To address this, it was proposed to combine economics and control into a single layer, and solve a dynamic optimization problem with economic cost function. This gives rise to economic MPC, which is explained in details in a book by Ellis et al. (2016).

As the dynamic process models in economic MPC become more complex, the online optimization problem cannot be solved sufficiently fast, which may result in instability of the closed loop system. To reduce the delay between obtaining a new measurement and implementing the inputs in the plant, fast sensitivity-based approaches were proposed. Two prominent methods are *real-time iteration* (RTI) (Diehl et al., 2002) and the *advanced-step NMPC* (Zavala and Biegler, 2009). These methods are based on the concept of parametric nonlinear programming (NLP) (Guddat et al., 1990), where the initial state values are considered as a parameter in the optimization problem.

Most sensitivity-based methods for NMPC assume strong regularity conditions on the NLP solution, typically the linear independence constraint qualification (LICQ), which guarantees unique multipliers. The more general case with non-unique multipliers was first addressed in Jäschke et al. (2014), who developed a pathfollowing NMPC procedure, under the condition that the Mangasarian-Fromovitz constraint qualification (MFCQ) holds. MFCQ may be satisfied during prediction horizon, for example, when path constraints are active for longer durations, a situation that occurs quite commonly in NMPC with economic cost functions (see also Vicente and Wright (2002) for a more general discussion).

The contribution of this work is to present an improved version of the path-following NMPC presented in Jäschke et al. (2014). In particular, we employ the path-following algorithm proposed in Kungurtsev and Jäschke (2017) to obtain approximate solutions to the dynamic optimization problem, and apply it in an asNMPC framework (Zavala and Biegler, 2009) in a similar manner as Suwartadi et al. (2017). The improved path-following method tracks the optimal solution along a parameter change by performing the following steps: first, a system of linear equations is solved as a Newton corrector step. This step refines the current primal and dual variables. Secondly, a quadratic programming (QP) problem is solved to find the directional derivative that is used as a predictor step. Finally, a linear program (LP) is solved to allow for jumps in the multipliers.

This paper is organized as follows. We formulate the NMPC problem in Section 2, and present the ideal NMPC that assumes zero computation time. The predictor-corrector path-following method for degenerate NLP (non-unique multipliers) is explained in Section 3, as well as its application in an advanced-step NMPC. We present our a case study in Section 4 and conclude the paper with a discussion and remarks in Section 5.

2. ECONOMIC NMPC

2.1 Ideal Economic NMPC

The economic MPC controller computes the optimal control input by solving the following optimization problem

$$\begin{aligned}\mathcal{P}_N\left(\mathbf{x}_k\right): \min_{\mathbf{z}_l,\mathbf{v}_l} \Psi\left(\mathbf{z}_N\right) &+ \sum_{l=0}^{N-1} \psi\left(\mathbf{z}_l,\mathbf{v}_l\right) \qquad (1)\\ \text{s.t.} \quad \mathbf{z}_{l+1} &= f\left(\mathbf{z}_l,\mathbf{v}_l\right), \quad l=0,\ldots,N-1\\ \mathbf{z}_0 &= \mathbf{x}_k,\\ \left(\mathbf{z}_l,\mathbf{v}_l\right) &\in \mathcal{Z}, \quad l=0,\ldots,N-1\\ \mathbf{z}_N &\in \mathcal{X}_f,\end{aligned}$$

where $\mathbf{z}_l \in \mathbb{R}^{n_z}$ and $\mathbf{v}_l \in \mathbb{R}^{n_v}$ are predicted state and control variables at sample time l, respectively. The economic objective function consists of the terminal cost $\Psi\left(\mathbf{z}_N\right) \in \mathcal{C}^2 : \mathbb{R}^{n_z} \to \mathbb{R}$ and the stage costs $\psi\left(\mathbf{z}_l,\mathbf{v}_l\right) \in \mathcal{C}^2 : \mathbb{R}^{n_z} \times \mathbb{R}^{n_v} \to \mathbb{R}$. The constraints include a discrete time dynamical system $f \in \mathcal{C}^2 : \mathbb{R}^{n_z} \times \mathbb{R}^{n_v} \to \mathbb{R}^{n_z}$, the equality constraint for initial condition $\mathbf{z}_0$, which is obtained from the measurement of the actual state $\mathbf{x}_k \in \mathbb{R}^{n_z}$ at the time instance k, and the final state variable $\mathbf{z}_N$ is contained within the set of terminal constraint $\mathcal{X}_f$. The set $\mathcal{Z}$ denotes the path constraint limiting the predicted state and control.

Having obtained a solution of the optimization problem $\mathcal{P}_N$, the first move of the optimized predicted control input $\mathbf{u}_k := \mathbf{v}_0$ is applied to the plant which evolves such that

$$\mathbf{x}_{k+1} = f\left(\mathbf{x}_k,\mathbf{u}_k\right), \qquad (2)$$

where $\mathbf{x}_k$ is the actual plant. When noise is present, the plant dynamics become

$$\mathbf{x}_{k+1} = f\left(\mathbf{x}_k,\mathbf{u}_k\right) + \mathbf{w}_k, \qquad (3)$$

where $\mathbf{w}_k \in \mathbb{R}^{n_z}$ represents the process noise. As the time $k = 1, 2, \ldots$ evolves, the optimization problem $\mathcal{P}_N$ is solved repeatedly in a receding horizon fashion as follows:

(1) Obtain measurement data $\mathbf{x}_k$,
(2) Solve the optimization problem $\mathcal{P}_N\left(\mathbf{x}_k\right)$,
(3) Inject the control input $\mathbf{u}_k := \mathbf{v}_0$ in the plant,
(4) Set $k \leftarrow k+1$, repeat from Step (1).

We refer to the procedure above as an *ideal NMPC* (*iNMPC*) controller.

2.2 Enforcing convergence for economic NMPC

Since the stage cost in the optimization problem $\mathcal{P}_N$ (1) can be any arbitrary economic measure, it may be difficult to ensure that the closed loop system is stable. See Faulwasser et al. (2018) for a detailed treatment of this topic. Stability may be ensured by regulating the NMPC problem such that the process approaches its steady-state optimal point, which can be found by solving

$$\begin{aligned}\min_{(\mathbf{x},\mathbf{u})\in\mathcal{Z}} \quad & \psi\left(\mathbf{x},\mathbf{u}\right) \qquad (4)\\ \text{s.t.}\ & \mathbf{x} - f\left(\mathbf{x},\mathbf{u}\right) = 0.\end{aligned}$$

Denoting the solution of (4) as $(\mathbf{x}_s,\mathbf{u}_s)$, we modify the economic NMPC stage cost in (1) by incorporating a regularization term so that it becomes

$$\psi_m\left(\mathbf{z},\mathbf{v}\right) := \psi\left(\mathbf{z},\mathbf{v}\right) + \alpha_{state}\left(\| \mathbf{z}-\mathbf{x}_s \|\right) + \alpha_{input}\left(\| \mathbf{v}-\mathbf{u}_s \|\right). \quad (5)$$

The value of the weights α_{state} and α_{input} may be chosen, for example, by the Gershgorin bound criteria, see Jäschke et al. (2014). Sufficiently large values of the weights guarantee that the resulting closed-loop system is asymptotically stable (Angeli et al., 2012).

2.3 The Advanced-step NMPC

The optimization problem (1) is the same from one MPC iteration to another, except for the changing measurement $\mathbf{x}_k$. Hence, the initial state variable may be considered a parameter. To reduce the computational time for solving the optimization problem $\mathcal{P}_N$ (1), instead of solving a full NLP problem, the asNMPC computes the sensitivity of NLP solution with respect to the initial a variable (parameter) $\mathbf{x}_k$. This can be used to obtain a first-order approximation of the solution at a nearby parameter. Based on NLP sensitivity, the asNMPC procedure includes the following three steps (Zavala and Biegler, 2009).

(1) (*Offline step*) Solve the NLP problem $\mathcal{P}_N\left(\mathbf{z}_{k+1}\right)$ offline at time k while setting the initial state value to the predicted state at $k+1$.
(2) (*Online step*) When the measurement $\mathbf{x}_{k+1}$ becomes available at time $k+1$, update the optimal solution obtained from the offline step using the sensitivity of the optimal solution from step 1.
(3) Implement the optimal control input and update $k \leftarrow k+1$ and repeat from Step 1.

If LICQ and strict complementarity hold together with a suitable second order condition, the optimal sensitivity update is step 2 can be calculated by solving a system of linear equations that can be formulated using the Karush-Kuhn-Tucker (KKT) system of the NLP.

The original asNMPC algorithm faces a challenge when an active-set change occurs along the parameter updates. In this case, the sensitivity must be computed by solving a quadratic program (Jäschke et al., 2014; Suwartadi et al., 2017) described in the next section, or by introducing heuristics such as "clipping in the first interval" (Biegler et al., 2015).

3. PREDICTOR-CORRECTOR PATH-FOLLOWING ECONOMIC NMPC

We explain the predictor-corrector path-following method in this section along with its application to an advanced-step economic NMPC controller. We define the following notation. The ith component of a vector $\mathbf{v}$ is denoted by $[\mathbf{v}]_i$ and if $\mathcal{K}$ is is an index set then $[\mathbf{v}]_{\mathcal{K}}$ represents the vector with $|\mathcal{K}|$ components composed of the entries of $\mathbf{v}$.

3.1 Preliminaries

We consider the optimization problem (1) as a parametric nonlinear optimization problem of the form

$$\begin{aligned} \min_{\boldsymbol{\chi}} \quad & F(\boldsymbol{\chi}, \mathbf{p}) \\ \text{subject to}\, & c_i(\boldsymbol{\chi}, \mathbf{p}) = 0, i \in \mathcal{E}, \\ & c_i(\boldsymbol{\chi}, \mathbf{p}) \leq 0, i \in \mathcal{I}, \end{aligned} \tag{6}$$

where $F : \mathbb{R}^{n_\chi} \times \mathbb{R}^{n_p} \to \mathbb{R}$ is the objective function, $\boldsymbol{\chi} \in \mathbb{R}^{n_\chi}$ is the primal variable and $\mathbf{p} \in \mathbb{R}^{n_p}$ is the parameter. The equality and inequality constraint sets are denoted by $\mathcal{E} = \{1, \ldots, m\}$ and $\mathcal{I} = \{m+1, \ldots, n\}$, respectively.

In the path-following algorithm, we track the optimal solution for a parameter change starting from $\mathbf{p}_0$ (here $\mathbf{z}_{k+1}$) to a final value $\mathbf{p}_f$ (here $\mathbf{x}_{k+1}$).

The Lagrangian is defined as

$$L(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) := F(\boldsymbol{\chi}, \mathbf{p}) + \mathbf{y}^T c(\boldsymbol{\chi}, \mathbf{p}), \tag{7}$$

where $\mathbf{y} \in \mathbb{R}^n$ is the dual variable. The Karush-Kuhn-Tucker (KKT) conditions for the problem are

$$\begin{aligned} \nabla_{\boldsymbol{\chi}} L(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) &= 0, \\ c_i(\boldsymbol{\chi}, \mathbf{p}) &= 0,\ i \in \mathcal{E}, \\ c_i(\boldsymbol{\chi}, \mathbf{p}) &\leq 0,\ i \in \mathcal{I}, \\ \mathbf{y}^T c(\boldsymbol{\chi}, \mathbf{p}) &= 0, \\ \mathbf{y}_i &\geq 0,\ i \in \mathcal{I}. \end{aligned} \tag{8}$$

We denote set of active inequality constraints as $\mathcal{A}(\boldsymbol{\chi}, \mathbf{p}) = \{c_i(\boldsymbol{\chi}, \mathbf{p}) = 0, i \in \mathcal{I}\}$. For a given multiplier $\mathbf{y}$ that satisfies (8) the active inequality set $\mathcal{A}(\boldsymbol{\chi}, \mathbf{p})$ has two subsets, which are the weakly active set $\mathcal{A}_0(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) = \{i \in \mathcal{A}(\boldsymbol{\chi}, \mathbf{p}) \mid \mathbf{y}_i = 0\}$ and a strongly active set $\mathcal{A}_+(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) = \{i \in \mathcal{A}(\boldsymbol{\chi}, \mathbf{p}) \mid \mathbf{y}_i > 0\}$. Furthermore, we use notation $\mathcal{A}_{+,j}$ to indicate strongly active set at iteration index j.

The Hessian of the Lagrangian with respect to the primal variables is

$$H(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) = \nabla^2_{\boldsymbol{\chi}\boldsymbol{\chi}} F(\boldsymbol{\chi}, \mathbf{p}) + \sum_{i=1}^{n} \nabla^2_{\boldsymbol{\chi}\boldsymbol{\chi}} c_i(\boldsymbol{\chi}, \mathbf{p}) \mathbf{y}_i. \tag{9}$$

Definition 1. Strong second-order sufficient conditions (SSOSC) holds at $(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p})$ if a pair of primal-dual variable $(\boldsymbol{\chi}, \mathbf{y})$ satisfies the first-order conditions (8) at $\mathbf{p}$ and

$$\mathbf{d}^T H(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) \mathbf{d} > 0 \, \forall \mathbf{d} \in \mathcal{C}(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) \setminus \{0\},$$

where the set $\mathcal{C}(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p})$ is defined as

$$\mathcal{C}(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) := \left\{\mathbf{d} : \nabla_{\boldsymbol{\chi}} c_i(\boldsymbol{\chi}, \mathbf{p})^T \mathbf{d} = 0 \, for \, i \in \mathcal{A}_+(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) \cup \mathcal{E}\right\}.$$

Definition 2. General Strong Second-order Sufficient Optimality Conditions (GSSOSC) is satisfied if the SSOSC is satisfied for all $\mathbf{y}$ that fulfill the first-order necessary conditions (8).

We require a constraint qualification to ensure that the KKT conditions (8) are a necessary condition for optimality. A standard constraint qualification that is frequently used is the linear independence constraint qualification (LICQ) which requires that the gradients of the active constraints are linearly independent. However, in dynamic optimization with path constraints, this may not be satisfied, see Vicente and Wright (2002). A constraint qualification that is more likely to hold in this case is the MFCQ.

Definition 3. Mangasarian-Fromovitz Constraint Qualification (MFCQ) holds at $(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p})$ if

(1) $\{\nabla_{\boldsymbol{\chi}} c_i(\boldsymbol{\chi}, \mathbf{p}), i \in \mathcal{E}\}$ is linearly independent,
(2) There exists a direction $\mathbf{s}$ such that $\nabla_{\boldsymbol{\chi}} c_i(\boldsymbol{\chi}, \mathbf{p})^T \mathbf{s} = 0$ for all $i \in \mathcal{E}$ and $\nabla_{\boldsymbol{\chi}} c_i(\boldsymbol{\chi}, \mathbf{p})^T \mathbf{s} < 0$ for all $i \in \mathcal{A}(\boldsymbol{\chi}, \mathbf{p})$.

The MFCQ implies that the set of multipliers that satisfy (8) is a bounded polytope (Gauvin, 1977).

Definition 4. The Constant Rank Constraint Qualification (CRCQ) holds at $(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p})$ if there exists a neighborhood $\mathcal{N}$ of $\boldsymbol{\chi}$ such that for all subsets $\mathfrak{U} \subseteq \mathcal{E} \cup \mathcal{A}(\boldsymbol{\chi}, \mathbf{p})$, the rank of $\{\nabla_{\boldsymbol{\chi}} c_i(\boldsymbol{\chi}, \mathbf{p}), i \in \mathfrak{U}\}$ is equal to the rank of $\{\nabla_{\boldsymbol{\chi}} c_i(\bar{\boldsymbol{\chi}}, \mathbf{p}), i \in \mathfrak{U}\}$ for all $\bar{\boldsymbol{\chi}} \in \mathcal{N}$.

Note that CRCQ is neither weaker nor stronger than MFCQ. Finally, we define the optimality residual as

$$\eta(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}) = \left\| \begin{pmatrix} \nabla_{\boldsymbol{\chi}} F(\boldsymbol{\chi}, \mathbf{p}) + \nabla_{\boldsymbol{\chi}} c(\boldsymbol{\chi}, \mathbf{p}) \mathbf{y} \\ c(\boldsymbol{\chi}, \mathbf{p})_{\mathcal{E}} \\ [\min(c(\boldsymbol{\chi}, \mathbf{p}), \mathbf{y})]_{\mathcal{I}} \end{pmatrix} \right\|_{\infty}, \tag{10}$$

which indicates how far a point $(\boldsymbol{\chi}, \mathbf{y}, \mathbf{p})$ is from a KKT point.

3.2 Predictor-Corrector Path-Following

The path-following method for tracing the optimal solution along a parameter change is described in Kungurtsev and Jäschke (2017) and consists of three steps: a corrector step, a predictor step, and a multiplier jump step. These three steps are run repeatedly to follow the path of optimal solutions, starting from initial parameter value $\mathbf{p}_0$ until final parameter $\mathbf{p}_f$. The parameter $\mathbf{p}$ is updated according to $\mathbf{p}(t_j) = (1 - t_j)\mathbf{p}_0 + t_j \mathbf{p}_f$, where $t_0 = 0$ until it reaches $t_j = 1$, that is $t_0 = 0 < t_1 < t_2 \ldots < t_j \ldots \leq 1$. We denote the primal and dual variables during the course of path-following iteration as $\boldsymbol{\chi}_j$ and $\mathbf{y}_j$ respectively, where j represents the index of the iteration along the path.

The three steps of the path-following algorithm are:

1.Corrector Step. This step takes an approximate solution of the primal variables and the strongly active dual variables and refines them for a given value of $\mathbf{p}$. This is done by solving the system of a linear equations

$$\boldsymbol{A} \begin{pmatrix} \Delta_c \boldsymbol{\chi} \\ \Delta_+ \mathbf{y} \end{pmatrix} = -\boldsymbol{B}, \tag{11}$$

where $\boldsymbol{A} = \begin{pmatrix} H(\boldsymbol{\chi}_j, \mathbf{y}_j, t) & \nabla_{\boldsymbol{\chi}} c_{\mathcal{A}_{+,j}}(\boldsymbol{\chi}_j, t) \\ \nabla_{\boldsymbol{\chi}} c_{\mathcal{A}_{+,j}}(\boldsymbol{\chi}_j, t)^T & 0 \end{pmatrix}$,

and $\boldsymbol{B} = \begin{pmatrix} \nabla_{\boldsymbol{\chi}} F(\boldsymbol{\chi}_j, t) + \nabla_{\boldsymbol{\chi}} c(\boldsymbol{\chi}_j, t) \mathbf{y}_j \\ \nabla_{\boldsymbol{\chi}} c_{\mathcal{A}_{+,j}}(\boldsymbol{\chi}_j, t) \end{pmatrix}$.

Since LICQ may not hold, the Jacobian $\nabla_{\boldsymbol{\chi}} c_{\mathcal{A}_{+,j}}(\boldsymbol{\chi}_j, t)$ is not full rank, unless the dual variables $\mathbf{y}_k$ are chosen from vertex of the polytope of feasible multipliers (Ralph and Dempe, 1995). This can be done by solving a linear program using a simplex method. Further below we will present a suitable linear program.

The approximate dual variables for the strongly active constraint are obtained from the solution (11), i.e., $[\Delta_c \mathbf{y}]_{\mathcal{A}_+,j} = \Delta_+\mathbf{y}$ and the remaining multipliers are set to zero, $[\Delta_c \mathbf{y}]_{\{1,\ldots,n\}\setminus\mathcal{A}_+,j} = 0$.

2.Predictor Step. Based on the improved solution from the corrector step, we compute a predictor by solving the following predictor QP

$$\min_{\Delta_p\boldsymbol{\chi}} \frac{1}{2}\Delta_p\mathcal{H}\boldsymbol{\chi}^T\Delta_p\boldsymbol{\chi} + \mathcal{F}^T\Delta_p\boldsymbol{\chi} \tag{12}$$
$$\begin{aligned} \text{subject to} \quad & \beta + \alpha^T\Delta_p\chi &= 0,\ i \in \mathcal{A}_{+,j} \\ & \beta + \alpha^T\Delta_p\chi &\le 0,\ i \in \mathcal{A}_j\setminus\mathcal{A}_{+,j} \end{aligned}$$

where

$$\mathcal{H} = H\left(\boldsymbol{\chi}_j, \mathbf{y}_j, t+\Delta t\right),$$
$$\mathcal{F} = \left(\nabla_{\boldsymbol{\chi}}F\left(\boldsymbol{\chi}_j, t+\Delta t\right) - \nabla_{\boldsymbol{\chi}}F\left(\boldsymbol{\chi}_j, t\right)\right),$$
$$\beta = \nabla_t c_i\left(\boldsymbol{\chi}_j, t\right)\Delta t,$$
$$\alpha = \left(\nabla_{\boldsymbol{\chi}}c_i\left(\boldsymbol{\chi}_j, t+\Delta t\right) + \nabla^2_{\boldsymbol{\chi}\boldsymbol{\chi}}c_i\left(\boldsymbol{\chi}_j, t+\Delta t\right)\Delta_c\boldsymbol{\chi}\right).$$

We obtain the primal and dual solution in this step $(\Delta_p\boldsymbol{\chi}, \Delta_p\mathbf{y})$. Combining this with the solution from the corrector step, we get $(\Delta\boldsymbol{\chi}, \Delta\mathbf{y}) = (\Delta_c\boldsymbol{\chi} + \Delta_p\boldsymbol{\chi}, \Delta_c\mathbf{y} + \Delta_p\mathbf{y})$. Here, we update the primal and dual variables solutions, i.e., $\boldsymbol{\chi}_{j+1} = \boldsymbol{\chi}_j + \Delta\boldsymbol{\chi}$, $\mathbf{y}_{j+1} = \mathbf{y}_j + \Delta\mathbf{y}$ and consequently the strongly active set $\mathcal{A}_{+,j+1}$.

3.Multiplier Jump Step. In order to allow for discontinuity in the multipliers along the path, we compute the dual variable solutions by solving the following LP,

$$\min_{\mathbf{y}} \mathbf{y}^T\nabla_t c\left(\boldsymbol{\chi}_j + \Delta\boldsymbol{\chi}, t+\Delta t\right)\Delta t \tag{13}$$
$$\begin{aligned} \text{subject to} \quad & -|\Omega| \le \vartheta \le |\Omega| \\ & \mathbf{y}_{\mathcal{I}} \ge 0 \\ & \mathbf{y}_{i\notin\mathcal{A}_{j+1}} = 0. \end{aligned}$$

where

$$\Omega = \nabla_{\boldsymbol{\chi}}L(\boldsymbol{\chi}_j + \Delta\boldsymbol{\chi}, \mathbf{y}_j + \Delta\mathbf{y}, t+\Delta t),$$
$$\vartheta = \nabla_{\boldsymbol{\chi}}F(\boldsymbol{\chi}_j + \Delta\boldsymbol{\chi}, t+\Delta t) + \sum_{i\in\mathcal{A}_{j+1}} \nabla_{\boldsymbol{\chi}}c_i\left(\boldsymbol{\chi}_j + \Delta\boldsymbol{\chi}, t+\Delta t\right)\mathbf{y}_i.$$

The solution $(\mathbf{y}_{LP})$ redefines the dual variable solutions $\mathbf{y}_{j+1} = \mathbf{y}_{LP}$ and the strongly active set $\mathcal{A}_{+,j+1} = \left\{i : [\mathbf{y}_{j+1}]_i > 0\right\}$. The three steps are summarized in Algorithm 1.

We include the predictor-corrector path-following method in the online step 2 of the asNMPC controller. This is shown in Algorithm 2.

4. NUMERICAL CASE EXAMPLE

The proposed method is tested and compared against the iNMPC controller. All simulations are done in `MATLAB` using `CasADi` algorithmic differentation tool (Andersson, 2013) version 3.2.0, which includes `IPOPT` (Wächter and Biegler, 2006) as NLP solver. We use `MINOS QP` (Murtagh

Algorithm 1 Predictor-corrector path-following method

Input: t, $\boldsymbol{\chi}$, $\mathbf{y}$ close to solution $(\boldsymbol{\chi}^*(t), \mathbf{y}^*(t))$ such that $\{\nabla_{\boldsymbol{\chi}}c_i(\boldsymbol{\chi}, t)\}_{\{i\in\mathcal{I}:\,\mathbf{y}_i>0\}\cup\mathcal{E}}$ is linearly independent, Δt, $\eta_{max} < 1$, and $\iota_{max} \in \mathbb{N}$.
Output: $\boldsymbol{\chi}$ and $\mathbf{y}$ at $\mathbf{p}_f$

```
1: function MFCQ_PC_PF($\boldsymbol{\chi}, \mathbf{y}, \mathbf{p}_0, \mathbf{p}_f, \Delta t$)
2:   Define parameter $\gamma$ satisfying $0 < \gamma < 1$.
3:   Define $\mathcal{A}_+$.
4:   Set $j \leftarrow 0$.
5:   Set $t_j = 0$.
6:   while $t_j < 1$ do
7:     Solve (CorrectStep) for $(\Delta_c\boldsymbol{\chi}, \Delta_+\mathbf{y})$.
8:     if for some $i \notin \mathcal{E}$, $[\mathbf{y}_{\mathcal{A}_+} + \Delta_+\mathbf{y}] < 0$ then
9:       Solve (1) using an NLP solver at $t_j$
10:      Set $\iota = 1$
11:    end if
12:    Solve (QPPredict) for $(\Delta_p\boldsymbol{\chi}, \Delta_p\mathbf{y})$.
13:    Set $(\Delta\boldsymbol{\chi}, \Delta\mathbf{y}) = (\Delta_p\boldsymbol{\chi}, \Delta_p\mathbf{y}) + (\Delta_c\boldsymbol{\chi}, \Delta_c\mathbf{y})$.
14:    Compute $\eta_{j+\Delta} := \eta\left(\boldsymbol{\chi}_j + \Delta\boldsymbol{\chi},\ \mathbf{y}_j + \Delta\mathbf{y},\ t_j + \Delta t\right)$.
15:    if $\eta_{j+\Delta} < \eta_{max}$ then
16:      $\boldsymbol{\chi}_{j+1} \leftarrow \boldsymbol{\chi}_j + \Delta\boldsymbol{\chi}$
17:      $\mathbf{y}_{j+1} \leftarrow \mathbf{y}_j + \Delta\mathbf{y}$
18:      $t_{j+1} \leftarrow t_j + \Delta t$
19:      $\mathbf{p}(t_j) = (1 - t_j)\mathbf{p}_0 + t_j\mathbf{p}_f$
20:      if $\eta_{j+\Delta} < \eta_j^{1+\gamma}$ then          ▷ very good step
21:        Increase $\Delta t$.
22:      end if
23:      Update $\mathcal{A}_+$.
24:      Solve (JumpLP) to redefine $\mathbf{y}_{j+1}$.
25:      Let $\mathcal{A}_+ = \{i : [\mathbf{y}_{j+1}]_i > 0\} \cup \mathcal{E}$.
26:    else
27:      if $\iota \ge \iota_{max}$ then
28:        Solve (1) using an NLP solver at $t_j$
29:        Set $\iota = 1$
30:        Go to line 7
31:      else
32:        Decrease $\Delta t$
33:        $\iota \leftarrow \iota + 1$
34:        Go to line 12
35:      end if
36:    end if
37:    $j \leftarrow j + 1$.
38:  end while
39:  Return $\boldsymbol{\chi}$
40: end function
```

Algorithm 2 Fast Economic pf-NMPC algorithm

Input: initial state $\mathbf{x}_0$ and stepsize Δt.
Output: The actual state $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, ...

```
1: for $k = 0, 1, 2, \ldots$ do
2:   $[\boldsymbol{\chi}^*, \mathbf{y}^*] \leftarrow$ solution of the NLP $\mathcal{P}_N(\mathbf{z}_{k+1})$ for
3:   predicted value $k + 1$.
4:   if a measurement of $\mathbf{x}_{k+1}$ is available then
5:     Set $\mathbf{p}_0 = \mathbf{z}_{k+1}$
6:     Set $\mathbf{p}_f = \mathbf{x}_{k+1}$
7:     $\boldsymbol{\chi}^* \leftarrow$ MFCQ_PC_PF($\boldsymbol{\chi}^*, \mathbf{y}^*, \mathbf{p}_0, \mathbf{p}_f, \Delta t$)
8:     Inject the first input move of $\boldsymbol{\chi}^*$ to the plant
9:     Update initial state $\mathbf{x}_0 \leftarrow \mathbf{x}_{k+1}$
10:    Set $k + 1 \leftarrow k$
11:  end if
12: end for
```

and Saunders, 1982) solver from TOMLAB and CPLEX as LP solver (IBM, 2017).

4.1 Process Description

We implement the pf-NMPC controller for a CSTR, taken from Diehl et al. (2011), with first order reaction $A \to B$ and slightly regularized objective function. The dynamic model, derived from mass balance, is

$$\begin{aligned} \frac{dc_A}{dt} &= \frac{Q}{V}(c_{Af} - c_A) - kc_A \qquad (14)\\ \frac{dc_B}{dt} &= \frac{Q}{V}(-c_B) - kc_A, \end{aligned}$$

where c_A and c_B are the concentration of components A and B, respectively. The rate constant is $k = 1.2 \frac{L}{mol\, minute}$, the reactor volume is $V = 10\, L$, and the feed concentration is $c_{Af} = 1 \frac{mol}{L}$. The control input is denoted by Q with unit $\frac{L}{minute}$ and the state variables are the concentration c_A and c_B. The economic objective function is

$$J = -Q, \qquad (15)$$

which is incorporated with the regularization term as in the equation (5)

$$\psi_m := J + (c_A - 0.5)^2 + (Q - 12)^2 . \qquad (16)$$

The bound constraints on the control and state are

$$\begin{aligned} 10 \le Q \le 20,\\ 0.49 \le c_B \le 1. \end{aligned}$$

Note that the second bound constraint is not included in Diehl et al. (2011) and becomes active at the steady-state solution. We run NMPC controllers with prediction horizon N = 50 for 100 minutes simulation time with sampling time 1 minute. We set initial $\Delta t = 0.5$ for the pf-NMPC controller and $\eta_{max} = 0.001$. Direct collocation is used to discretize the optimal control problem, yielding 452 optimization variables and 402 nonlinear equality constraints from the discretized state equations (14) as well as bound constraints for the control input and state variables. The real plant model is simulated by 'ode15' solver in MATLAB. We add noise to the state measurements, where the noise is taken to have a normal distribution with zero mean and a variance of one percent of the steady state values.

We check the linear independence of the Jacobian of the active bound constraints and equality constraints at the solution of the open loop problem (1) and we find that the Jacobian is rank deficient, which implies LICQ is not satisfied. Another way to check, if LICQ does not hold, is that the the number of active constraints (equality constraints plus active bound constraints) exceeds the number of optimization variables, as one can observe from Figure 1.

We continue to check whether MFCQ holds. As in Definition 3 (MFCQ), there must exist a strictly feasible step **s** that satisfies the inequality constraints in addition to the linear independence of the equality constraints. The existence of the step **s** can be verified by solving an LP problem (Forsgren et al., 2002, Section 2.2). Another simpler

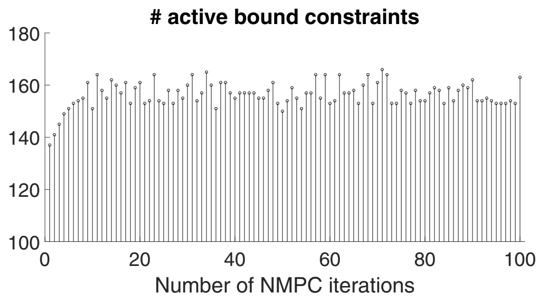

Fig. 1. Number of active bound constraints during the open-loop optimizations. The active constraints correspond to the lower bound on c_B.

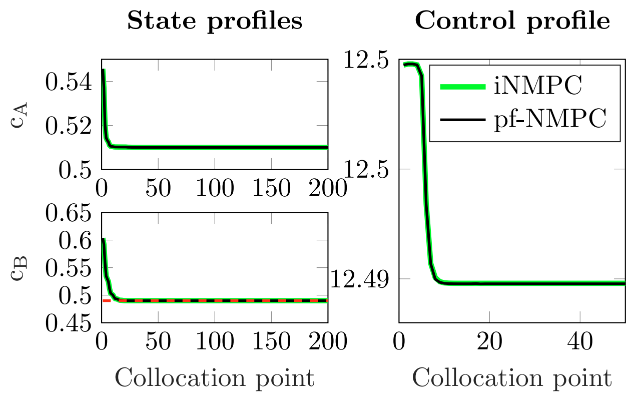

Fig. 2. Open loop solutions (state variables and control input) comparison at iteration number 2 for iNMPC and pf-NMPC controllers. The red color line in c_B figure denotes the lower bound constraint.

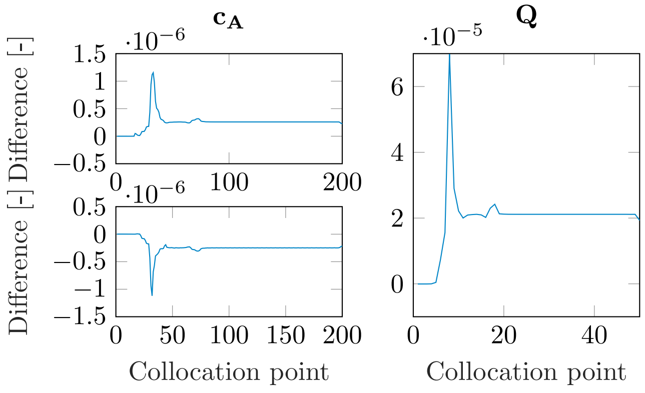

Fig. 3. The difference in predicted state variables and control input between iNMPC and pf-NMPC at iteration number 2.

approach can be employed in this case. It can be verified that for suitable initial conditions there exists a feasible control input such that $c_B > 0.49$ and $10 \le Q \le 20$. Also, the system has been setup such that the model equations (equality constraints) are linearly independent.

4.2 Comparison of Open-loop Optimization Results

Here, we compare open loop optimization solutions (predicted state and control) at MPC iteration number two. Due to the added noise, the prediction is not perfect, and is updated using the path-following algorithm. The

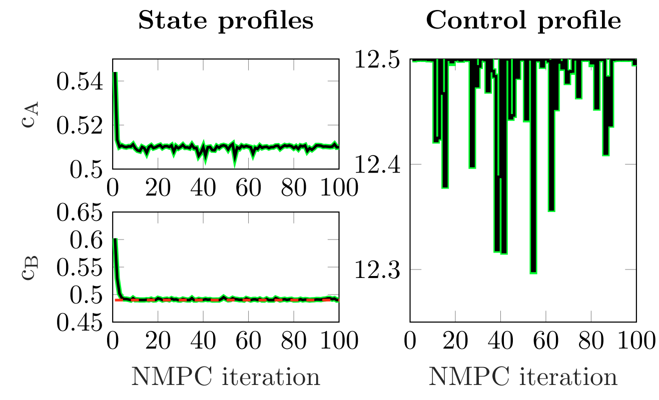

Fig. 4. Closed-loop solutions (state variables and control input) comparison of iNMPC and pf-NMPC controllers.

Table 1.

online optimization runtime (in sec.)			
	min	max	average
iNmpc	0.0680	0.1191	0.0884
pf-Nmpc	0.0416	0.1022	0.0479

results are depicted in Figure 2 in which the solutions of both controllers are plotted. The resulted trajectories of state and control inputs of pf-NMPC follow precisely the iNMPC solutions. The differences between the solutions are shown in Figure 3. The differences in the state variables and control input are in the order of 10^{-7} and 10^{-5}, respectively. This shows that the path-following algorithm tracks the true NLP solution very accurately.

4.3 Closed-loop Results

We compare the closed-loop responses of pf-NMPC controller, which are obtained from the plant measurement data after injecting the first move of the optimized control input. As can be seen in Figure 4, again, the pf-NMPC solutions track accurately those of iNMPC. Furthermore, we compare online runtime between iNMPC and pf-NMPC updates in Table 1, where on average our pf-NMPC approach gives a speedup factor of almost two. However, these numbers are very implementation dependent, and should only be used as an indication. A detailed comparison is not the scope of this work.

5. CONCLUSION

We have proposed the use of a predictor-corrector path-following method, consisting of the three steps (corrector, predictor, and multiplier jump step), for solving the online open-loop optimal control problem in an economic NMPC. We have shown that the pf-NMPC works as expected in the case example, and accurately tracks the solutions of iNMPC controller, in the presence of non-unique multipliers.

REFERENCES

Andersson, J. (2013). *A general-purpose software framework for dynamic optimization*. Ph.D. thesis, Arenberg Doctoral School, KU Leuven.

Angeli, D., Amrit, R., and Rawlings, J.B. (2012). On average performance and stability of economic model predictive control. *IEEE transactions on automatic control*, 57(7), 1615–1626.

Biegler, L., Yang, X., and Fischer, G. (2015). Advances in sensitivity-based nonlinear model predictive control and dynamic real-time optimization. *Journal of Process Control*, 30, 104–116.

Diehl, M., Amrit, R., and Rawlings, J.B. (2011). A lyapunov function for economic optimizing model predictive control. *IEEE Transactions on Automatic Control*, 56(3), 703–707.

Diehl, M., Bock, H.G., Schlöder, J.P., Findeisen, R., Nagy, Z., and Allgöwer, F. (2002). Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4), 577–585.

Ellis, M., Liu, J., and Christofides, P.D. (2016). *Economic Model Predictive Control: Theory, Formulations and Chemical Process Applications*. Springer.

Faulwasser, T., Grüne, L., Müller, M.A., et al. (2018). Economic nonlinear model predictive control. *Foundations and Trends® in Systems and Control*, 5(1), 1–98.

Forsgren, A., Gill, P.E., and Wright, M.H. (2002). Interior methods for nonlinear optimization. *SIAM review*, 44(4), 525–597.

Gauvin, J. (1977). A necessary and sufficient regularity condition to have bounded multipliers in nonconvex programming. *Mathematical Programming*, 12(1), 136–138.

Guddat, J., Vazquez, F.G., and Jongen, H.T. (1990). *Parametric optimization: singularities, pathfollowing and jumps*. Springer.

IBM (2017). *IBM ILOG CPLEX Optimizers 12.7.1.*

Jäschke, J., Yang, X., and Biegler, L.T. (2014). Fast economic model predictive control based on nlp-sensitivities. *Journal of Process Control*, 24(8), 1260–1272.

Kungurtsev, V. and Jäschke, J. (2017). A predictor-corrector path-following algorithm for dual-degenerate parametric optimization problems. *SIAM Journal on Optimization*, 27(1), 538–564.

Murtagh, B.A. and Saunders, M.A. (1982). A projected lagrangian algorithm and its implementation for sparse nonlinear constraints. *Algorithms for Constrained Minimization of Smooth Nonlinear Functions*, 84–117.

Ralph, D. and Dempe, S. (1995). Directional derivatives of the solution of a parametric nonlinear program. *Mathematical programming*, 70(1-3), 159–172.

Suwartadi, E., Kungurtsev, V., and Jäschke, J. (2017). Sensitivity-based economic nmpc with a path-following approach. *Processes*, 5(1), 8.

Vicente, L.N. and Wright, S.J. (2002). Local convergence of a primal-dual method for degenerate nonlinear programming. *Computational Optimization and Applications*, 22(3), 311–328.

Wächter, A. and Biegler, L.T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1), 25–57.

Zavala, V.M. and Biegler, L.T. (2009). The advanced-step nmpc controller: Optimality, stability and robustness. *Automatica*, 45(1), 86–93.