Journal of Process Control 44 (2016) 162–183

Contents lists available at ScienceDirect

Journal of Process Control

journal homepage: www.elsevier.com/locate/jprocont

Review

Fast NMPC schemes for regulatory and economic NMPC – A review[☆]

Inga J. Wolf, Wolfgang Marquardt[*]

AVT – Process Systems Engineering, RWTH-Aachen University, Aachen, Germany

ARTICLE INFO

Article history:
Received 15 April 2015
Received in revised form 9 March 2016
Accepted 6 May 2016
Available online 16 June 2016

Keywords:
Nonlinear model-predictive control
Economic nonlinear model-predictive control
Fast update methods
Suboptimal NMPC
Real-time iterations
Neighboring-extremal control

ABSTRACT

In this paper, NMPC schemes based on fast update methods (fast NMPC schemes) are reviewed that strive to provide a fast but typically suboptimal update of the control variables at each sampling instant with negligible computational delay. The review focuses on schemes that employ one of two subclasses of fast update methods developed for direct solution approaches, the suboptimal update methods and the sensitivity-based update methods. The connections and similarities of the fast update methods, the elements of the fast NMPC, the control architecture as well as the fast NMPC schemes as a whole are highlighted to support the assessment of the benefits and limitations of each individual scheme. In this way, this review facilitates the choice of a suitable fast NMPC scheme within the vast amount of fast NMPC schemes available in literature.

Contents

1. Introduction 163
2. Preliminaries 164
    2.1. Open-loop optimal control problem 164
    2.2. Nonlinear parametric programming problem 164
3. Fast update methods for PNLP 165
    3.1. Suboptimal update methods 165
    3.2. Sensitivity-based update methods 165
        3.2.1. Constant active set 165
        3.2.2. Changing active set 166
    3.3. Comparison of fast update methods 167
        3.3.1. Constant active set 168
        3.3.2. Changing active set 168
        3.3.3. Summary 168
    3.4. Illustrative example 169
        3.4.1. Optimal initial point and constant active set 169
        3.4.2. Optimal initial point and changing active set 169
        3.4.3. Suboptimal initial point and changing active set 169
    3.5. Reduction of computational time for fast update methods 169
    3.6. Relation to nonlinear parametric programming 169
    3.7. Discussion 171
4. Fast NMPC schemes tailored to a single-layer control architecture 171
    4.1. Offline preparations 172
    4.2. Operational phases of centralized controller 172

☆ A shortened review based on the first submitted version of this contribution has been published in [1].
* Corresponding author. Present affiliation: Forschungszentrum Jülich GmbH in the Helmoltz Association, D-52425 Jülich, Germany.
E-mail address: w.marquardt@fz-juelich.de (W. Marquardt).

http://dx.doi.org/10.1016/j.jprocont.2016.05.002
0959-1524/© 2016 Elsevier Ltd. All rights reserved.

4.2.1. Online preparation phase 172
4.2.2. Feedback phase 173
4.3. Fast NMPC schemes in literature 173
4.3.1. Real-time optimization by Bock et al. [42] 173
4.3.2. Real-time iterations by Diehl et al. [18] 173
4.3.3. Neighboring-extremal controller by Würth et al. [17] 173
4.3.4. Advanced-step controller by Zavala and Biegler [21] 174
4.3.5. NMPC with incomplete optimization by Grüne and Pannek [66] 174
4.3.6. Real-time nonlinear optimization as a generalized equation by Zavala and Anitescu [35] 174
4.3.7. Best optimal control performance by Wolf et al. [34] 174
4.3.8. Fast economic model predictive control by Jäschke et al. [28] 174
4.4. Case study 174
4.5. Discussion 175
5. Extensions to fast NMPC schemes 176
5.1. Time-scale separation 176
5.1.1. A two-layer fast NMPC scheme by Kadam and Marquardt [31] 176
5.1.2. Real-time iterations on multiple layers by Bock et al. [13] 176
5.1.3. A two-layer fast NMPC scheme by Würth et al. [72] 177
5.1.4. A two-layer fast NMPC scheme by Wolf et al. [8] 177
5.1.5. Hierarchical distributed neighboring-extremal controller by Wolf et al. [36] 177
5.1.6. Serial approach of the advanced-multi-step controller by Yang and Biegler [29] 177
5.2. Parallel computing 177
5.2.1. Parallel approach of the advanced-multi-step controller by Yang and Biegler [29] 177
5.3. Case study 177
5.3.1. First scenario: fluctuating reaction constant 177
5.3.2. Second scenario: jump in reaction constant 177
5.3.3. Third scenario: trend in reaction constant 178
5.4. Discussion 178
6. Closely related NMPC schemes 178
6.1. NMPC schemes based on linearization of model 179
6.1.1. Newton-type control strategy by Li and Biegler [75] 179
6.1.2. Nonlinear quadratic dynamic matrix control by Gattu and Zafiriou [77] 179
6.1.3. Constrained receding horizon predictive control by Lee et al. [78] 179
6.1.4. Model predictive control algorithm by Tiagounov and Weiland [79] 179
6.2. NMPC schemes based on fast update method for indirect solution approach 179
6.2.1. Neighboring extremals by Pesch [82] 179
6.2.2. Suboptimal MPC by Graichen and Kugi [67] 179
6.3. Explicit MPC 179
6.3.1. Explicit LMPC by Pistikopoulos et al. [90] 180
6.3.2. Explicit NMPC by Johansen [98] 180
6.3.3. Partial enumeration by Pannocchia et al. [91] 180
6.4. NMPC schemes based on continuation 180
6.4.1. Continuation/generalized minimum residual (GRMES) update method by Ohtsuka [101] 180
6.4.2. Real-time framework by DeHaan and Guay [106] 180
7. Conclusions 180
References 181

1. Introduction

Today's highly competitive and fluctuating markets call for the economically optimal operation of nonlinear processes at any time. This requirement can be met by nonlinear model-predictive control (NMPC), an advanced optimal control method, where an optimal control problem with a nonlinear objective function is repeatedly optimized online over a given time horizon subject to a nonlinear process model and nonlinear constraints. Nowadays, NMPC can be further subdivided into standard regulatory NMPC and the emerging economic NMPC (see [2,3]). While in regulatory NMPC the nonlinear objective function penalizes the deviation from a given setpoint or from reference trajectories, the objective function in economic NMPC represents an economic criterion. Despite its ability to naturally handle multi-input multi-output systems as well as constraints, NMPC is not yet as widely applied in industry as linear model-predictive control (see, for example, the survey paper of Qin and Badgwell [4]). Besides theoretical issues regarding major properties of NMPC like feasibility, stability and robustness, one of the main hindrances is that the computational time to solve the nonlinear optimal control problem to convergence is usually not negligible. This holds especially for large-scale processes. Any computational delay might result in a loss of control performance and even in instability [5]. This is particularly true if the ratio of computational time to sampling period is higher than one, since new information on the current state of the system cannot be taken into account.

To overcome this drawback, a vast amount of NMPC schemes[1] has been developed comprising, among others, hierarchical NMPC schemes (see the excellent reviews [4,6] and, for example, [7,8] for some recent publications), distributed NMPC schemes (see [6]) and NMPC schemes based on triggered evaluations (see, for example, [9,10]). Especially over the last 15 years, NMPC schemes have evolved that are based on fast update methods providing a fast update at each sampling instant. The fast update approximates the solution to a parametric nonlinear program (PNLP), which is obtained when the optimal control problem is discretized in a direct

[1] The term *NMPC scheme* comprises the control architecture, the individual NMPC-related controllers as well as the algorithms implemented in the controllers.

solution approach, with only few computations. Consequently, fast update methods may considerably reduce computational time such that it often becomes negligible in contrast to iterative solution procedures solving the PNLP to convergence at each sampling instant. Furthermore, these methods are theoretically backed by the seminal results that recursive feasibility alone suffices to guarantee stability for regulatory NMPC [11] and boundedness for economic NMPC [12]. In this work, we will focus on NMPC schemes, where at least one of the controllers is a fast nonlinear-model predictive controller (NMPC) providing a fast update of the control variables based on a fast update method developed for direct solution approaches. These schemes will be termed fast NMPC schemes in the following.

Though individual fast NMPC schemes rely on the theory of different research fields, their control architectures are often identical and they often share common elements regarding the fast NMPC applied. In particular, the fast update methods, which are applied in the fast NMPC to compute a fast update based on the current state of the system, are often related. These connections are not evident due to the numerous algorithms developed. Thus, the assessment of the benefits and limitations of each method in comparison to others is difficult. For this reason, this work reviews and assesses the common elements of the fast NMPC including fast update methods and the control architectures applied such that the choice of a suitable fast NMPC scheme is facilitated.

The paper is structured as follows. In Section 2, we reformulate the optimal control problem to a PNLP and state the corresponding Karush–Kuhn–Tucker (KKT) system. In Section 3, several fast update methods are introduced and assessed that belong to either one of two subclasses: while suboptimal update methods simply terminate the iterations of the iterative solution procedure before convergence is reached, sensitivity-based update methods employ a linearization of the KKT system with respect to changes in the parameters as a fast update. Section 4 deals with fast NMPC schemes that either rely on a suboptimal or a sensitivity-based update method. Common elements of the fast NMPC are identified which are used to provide fast updates. Moreover, several fast NMPC schemes are assessed with the help of a case study. In Section 5, extensions to the fast NMPC schemes are introduced which often address the case when the optimal solution to PNLP cannot be computed within one sampling period. Section 6 reviews a selection of closely related NMPC schemes.

2. Preliminaries

2.1. Open-loop optimal control problem

The continuous-time optimal control problem can be formulated as

$$\min_{u,t_f} \hat{\Phi}(t_f), \tag{1a}$$

$$s.t. \quad \dot{x}(t) = f(x(t), z(t), u(t)), \tag{1b}$$

$$0 = g(x(t), z(t), u(t)), \tag{1c}$$

$$x(t_0) = x_0, \tag{1d}$$

$$\hat{c}(x(t), z(t), u(t)) \preceq 0, \tag{1e}$$

$$\tilde{c}(x(t_f), t_f) \preceq 0, \tag{1f}$$

$$\bar{c}(x(t_f), t_f) = 0, \tag{1g}$$

$$t \in \mathcal{I} := [t_0, t_f], \tag{1h}$$

where $x : \mathcal{I} \to \mathbb{R}^{n_x}$ and $z : \mathcal{I} \to \mathbb{R}^{n_z}$ represent the trajectories of the differential and algebraic variables on time horizon $\mathcal{I}$, respectively. Furthermore, $u : \mathcal{I} \to \mathbb{R}^{n_u}$ represents the trajectories of the control variables. The objective function $\hat{\Phi}$ of the optimal control problem is of Mayer-type and subject to the differential-algebraic equation (DAE) model of index one, (1b) and (1c), with consistent initial conditions (1d) as well as vectors of path and endpoint constraints, (1e)–(1g), where the symbol $\preceq$ denotes component-wise inequality. Note that a regulatory or economic objective of the more general Bolza-type can be reformulated to Mayer-type by extending the DAE model with an additional differential variable for the Lagrange term. For notational simplicity, a single-stage formulation is chosen in (1), though it can be readily extended to a multi-stage formulation (cf. [8] for fast economic NMPC). $\hat{\Phi}, f, g, \hat{c}, \tilde{c}$ and $\bar{c}$ are assumed to be at least twice continuously differentiable. Furthermore, we assume that a feasible solution to problem (1) exists which is unique.

In (1), time-invariant uncertain parameters $\theta \in \mathbb{R}^{n_\theta}$ and time-variant disturbances $d : \mathcal{I} \to \mathbb{R}^{n_d}$ can also be included by introducing suitable differential variables in the model equations. In this way, all parameters that are subject to changes and that cannot be influenced directly can be cast into the initial conditions [13], which have to be updated online based on measurements or some estimation algorithm.

2.2. Nonlinear parametric programming problem

An approximation of the optimal solution to (1) can be obtained by time-discretization applying single shooting, multiple shooting or full discretization by collocation, for example. In this work, the different solution approaches will not be reviewed given the excellent recent contributions of Diehl et al. [14] and Biegler [15]. We rather focus on different fast update methods, which can be employed for any of these approaches. All these direct solution approaches discretize (1) such that the parametric nonlinear program (PNLP)

$$\begin{aligned} \min_{\zeta} \quad & \Phi(\zeta, p), \\ s.t. \quad & e(\zeta, p) = 0, \\ & i(\zeta, p) \preceq 0 \end{aligned} \tag{2}$$

is obtained, where $\zeta \in \mathbb{R}^{n_\zeta}$ denotes the decision variables and $p \in \mathbb{R}^{n_p}$ the parameters which correspond to the initial conditions in problem (1). $e : \mathbb{R}^{n_\zeta} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_e}$ are the functions related to the equality constraints and $i : \mathbb{R}^{n_\zeta} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_i}$ are the functions related to the inequality constraints. PNLP (2) can be solved by an iterative solution procedure such as sequential quadratic programming (SQP) starting from an initial guess ζ^{init} [16].

To later compare fast update methods, the KKT system of PNLP (2) will be briefly stated. We introduce the Lagrange multipliers $\mu \in \mathbb{R}^{n_e}$ and $\lambda \in \mathbb{R}^{n_i}$ for the equality constraints and the inequality constraints, respectively. The Lagrange function is then given by $L = \Phi + e^T\mu + i^T\lambda$. Suppose that ζ^* is the local optimal solution and that the linear independence constraint qualification (LICQ) hold, then the KKT system of PNLP (2) at $(\zeta, \mu, \lambda)^*$ is given by

$$\begin{aligned} & L_\zeta(\zeta^*, \mu^*, \lambda^*, p) = 0, \\ & e(\zeta^*, p) = 0, \\ & i(\zeta^*, p) \preceq 0, \\ & \lambda^{*T} i(\zeta^*, p) = 0, \\ & -\lambda^* \preceq 0. \end{aligned} \tag{3}$$

Here, the symbols Υ_w and $\Upsilon_{w\bar{w}}$ represent first- and second-order derivatives for a quantity Υ with respect to w and $\bar{w}$. If Υ is a scalar, Υ_w is a row vector.

Note that, alternatively to (2), the PNLP

$$\begin{aligned} \min_{\zeta,\tilde{\zeta}} \quad & \Phi(\zeta,\tilde{\zeta}), \\ s.t. \quad & e(\zeta,\tilde{\zeta}) = 0, \\ & p - \tilde{\zeta} = 0, \\ & i(\zeta,\tilde{\zeta}) \preceq 0 \end{aligned}$$

is often used in multiple shooting and full discretization approaches, where $\tilde{\zeta} \in \mathbb{R}^{n_p}$ are additional decision variables. As the (local) optimal decision variables ζ^* of one formulation are also the (local) optimal decision variables of the other and vice versa [17], we will solely consider PNLP (2) with its corresponding KKT system (3) in the following.

3. Fast update methods for PNLP

Fast update methods provide a fast approximation of the optimal solution to PNLP (2) under parametric changes. Consequently, they constitute a core element of any fast NMPC scheme. In this section, we first review fast update methods that either belong to the suboptimal or to the sensitivity-based update methods. Like iterative solution procedures start from an initial guess ζ^{init} to solve PNLP (2), these fast update methods require an initial point $(\zeta, \mu, \lambda)^{init}$ for the initial parameter vector p^{init} to provide a fast update. This initial point typically does not fulfill the KKT conditions (3) for p^{init} and is thus suboptimal. Otherwise, it is optimal and will be denoted by $(\zeta, \mu, \lambda)^{init^*}$. For the initial point, the functions e^{init} and i^{init} related to the equality and inequality constraints must be provided as well as the first- and second-order derivatives, Φ_ζ^{init}, e_ζ^{init}, i_ζ^{init}, e_p^{init}, i_p^{init}, $L_{\zeta\zeta}^{init}$, $L_{\zeta p}^{init}$ and L_{pp}^{init}, with respect to the decision variables ζ and/or parameter vector p. Here, the notation Υ^{init} denotes a function or matrix Υ evaluated at $(\zeta, \mu, \lambda)^{init}$ for p^{init}. For simplification of the argument, efficient approximations of the function values and derivatives as well as other important algorithmic details of QP-based fast update methods such as line search and trust-region methods will be neglected. After the review, the fast update methods are compared and an illustrative example is given. Then, approaches to reduce the computational time of a fast update are briefly stated. Finally, the relation of the fast update methods to nonlinear parametric programming is depicted and the results of this section are discussed.

3.1. Suboptimal update methods

The first class of fast update methods for PNLP (2) are the suboptimal update methods. Typically, the setting of SQP is employed for the solution of (2) for a fixed p^{init}. A suboptimal update of the solution to (2) can be obtained by a QP iteration starting from a given initial point $(\zeta, \mu, \lambda)^{init}$ for a fixed p^{init}. The QP update in an SQP algorithm is given by

$$\begin{aligned} \min_{\Delta\zeta} \quad & \Phi_\zeta^{init}\Delta\zeta + 0.5\Delta\zeta^T L_{\zeta\zeta}^{init}\Delta\zeta, \\ s.t. \quad & e^{init} + e_\zeta^{init}\Delta\zeta = 0, \\ & i^{init} + i_\zeta^{init}\Delta\zeta \preceq 0, \end{aligned} \tag{4}$$

where $\Delta\zeta = \zeta - \zeta^{init}$. Note that changes in the initial parameter vector p^{init} are not directly accounted for by the QP. Though solely one QP iteration is usually applied, also several QP iterations can be conducted to improve the suboptimal solution.

Diehl et al. [18] suggested so-called real-time iterations (RTI) for a receding horizon setting, where changes in the parameter vector $\Delta p = p - p^{init}$ arise when moving from one horizon to the next. RTI corresponds to the alternative QP

$$\begin{aligned} \min_{\Delta\zeta} \quad & \begin{bmatrix} \Phi_\zeta \, \Phi_p \end{bmatrix}^{init} \begin{bmatrix} \Delta\zeta \\ \Delta p \end{bmatrix} + 0.5 \begin{bmatrix} \Delta\zeta \\ \Delta p \end{bmatrix}^T \begin{bmatrix} L_{\zeta\zeta} & L_{\zeta p} \\ L_{p\zeta} & L_{pp} \end{bmatrix}^{init} \begin{bmatrix} \Delta\zeta \\ \Delta p \end{bmatrix} \\ s.t. \quad & e^{init} + e_\zeta^{init}\Delta\zeta + e_p^{init}\Delta p = 0, \\ & i^{init} + i_\zeta^{init}\Delta\zeta + i_p^{init}\Delta p \preceq 0, \end{aligned} \tag{5}$$

where Δp is provided. In contrast to (4), the alternative QP (5) accounts for changes in the initial parameter vector p^{init} besides improving a suboptimal initial point $(\zeta, \mu, \lambda)^{init}$.

3.2. Sensitivity-based update methods

The second class of fast update methods are the sensitivity-based update methods which provide an estimate of the optimal solution to PNLP (2) under parametric variations. This estimate is based on a first-order correction of the KKT system (3) with respect to p. In general, we can distinguish between sensitivity-based update methods designed for a constant and a changing active set of inequality constraints, respectively. Here, the inequality constraints can be divided into a set of strongly active constraints defined by $A = \{j | i_j = 0, \lambda_j > 0\}$, a set of inactive constraints defined by $I = \{j | i_j < 0, \lambda_j = 0\}$ and a set of weakly active constraints defined by $K = \{j | i_j = 0, \lambda_j = 0\}$. For a constant active set, the sets A, I and K remain the same before and after the update. In Section 3.2.1, a sensitivity-based update method is presented for a constant active set while, in Section 3.2.2, methods for a changing active set are reviewed.

3.2.1. Constant active set

The sensitivity-based update method introduced by Fiacco and McCormick [19] and Fiacco [20] assumes an optimal initial point $(\zeta, \mu, \lambda)^{init^*}$ for the initial parameter vector p^{init} and a constant active set of inequality constraints. According to the authors, a once continuously differentiable function $[\zeta(p)^T, \mu(p)^T, \lambda(p)^T]^T$ satisfying the second-order sufficient conditions of optimality exists in the neighborhood of $(\zeta, \mu, \lambda)^{init^*}$, if

C1. all functions in (2) are twice continuously differentiable in a neighborhood of $(\zeta, \mu, \lambda)^{init^*}$,
C2. the gradients of the constraints are linearly independent at $(\zeta, \mu, \lambda)^{init^*}$ for p^{init},
C3. the second-order sufficient conditions (SOSC) hold at $(\zeta, \mu, \lambda)^{init^*}$ and
C4. the strict complementarity condition holds at $(\zeta, \mu, \lambda)^{init^*}$, i.e., K is empty and the sets A and I remain unchanged.

Under these conditions, the derivative of the once continuously differentiable function with respect to p can be obtained by the implicit function theorem,

$$\begin{bmatrix} \dfrac{d\zeta}{dp} \\ \dfrac{d\mu}{dp} \\ \dfrac{d\lambda^a}{dp} \end{bmatrix}^{init*} = -\mathcal{M}^{init*,-1} \begin{bmatrix} L_{\zeta p} \\ e_p \\ (i^a)_p \end{bmatrix}^{init*}, \tag{6}$$

where i^a is the vector of functions related to the active inequality constraints, λ^a are the corresponding Lagrange multipliers and

$$\mathcal{M} := \begin{bmatrix} L_{\zeta\zeta} & e_\zeta^T & i_\zeta^{a,T} \\ e_\zeta & 0 & 0 \\ i_\zeta^a & 0 & 0 \end{bmatrix}.$$

For a constant active set, the sensitivity-based update (SBU) of an optimal initial point $(\zeta, \mu, \lambda)^{init^*}$ for p^{init} can be computed by

$$\begin{bmatrix} \zeta \\ \mu \\ \lambda^a \end{bmatrix} = \begin{bmatrix} \zeta \\ \mu \\ \lambda^a \end{bmatrix}^{init*} + \begin{bmatrix} \frac{d\zeta}{dp} \\ \frac{d\mu}{dp} \\ \frac{d\lambda^a}{dp} \end{bmatrix}^{init*} \Delta p, \tag{7}$$

where $\Delta p = p - p^{init}$ again accounts for changes in the initial parameter vector p^{init}.

3.2.2. Changing active set

If a change in the active set occurs as a consequence of the update, condition C4 is violated. To prevent this, the inequality constraints might be considered implicitly in the objective function as penalty terms (see, for example, Zavala and Biegler [21]). Because of the inevitable inaccuracy introduced by penalty methods (see Diehl et al. [14]), we will solely focus on fast update methods that handle a change in the active set explicitly. In the following, we will distinguish between methods which are derived for an optimal initial point and methods which include correction terms for a suboptimal initial point.

3.2.2.1. Methods assuming an optimal initial point. Already in the seventies, Bigelow and Shapiro [22] used a first-order correction of (3) with respect to p to describe a directionally differentiable function $[\zeta(p)^T, \mu(p)^T, \lambda(p)^T]^T$ in a neighborhood of the initial point $(\zeta, \mu, \lambda)^{init^*}$. At $(\zeta, \mu, \lambda)^{init^*}$, only conditions C1 to C3 have to hold. Though currently active and inactive constraints are not allowed to change their set, because $\lambda^T i^{init^*} = 0$ would no longer be fulfilled, the constraints in K can become active or inactive. The authors formulated the QP

$$\begin{aligned} \min_{\Delta\zeta} \quad & 0.5\Delta\zeta^T L_{\zeta\zeta}^{init*}\Delta\zeta + \Delta\zeta^T L_{\zeta p}^{init*}\Delta p, \\ s.t. \quad & e_\zeta^{init*}\Delta\zeta + e_p^{init*}\Delta p = 0, \\ & i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p = 0, \quad \forall j \in A, \\ & i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p \le 0, \quad \forall j \in K, \end{aligned} \tag{8}$$

determining the active set of constraints, where the Lagrange multiplier for constraint i_j^{init*} is $\Delta\lambda_j = \lambda_j - \lambda_j^{init*}$ and the multipliers for constraints e^{init^*} are $\Delta\mu = \mu - \mu^{init^*}$. Note that, though the authors did not consider equality constraints explicitly, they were added in (8) for completeness. Later, Shapiro [23] extended this result by both replacing the LICQ formulated in condition C2 with the more general Mangasarian–Fromovitz constraint qualifications (MFCQ) and by replacing the SOSC of condition C3 with the generalized second-order sufficient conditions.

In order to consider more general changes of the active set, Beltracchi and Gabriele [24] formulated a path-following algorithm (denoted as PFA1 subsequently) which is based on the determination of the so-called sensitivity domain, the domain in the parameter space $\mathbb{R}^{n_p}$, where the active set stays constant [25]. The parameter vector value p_k, $k = 1, \ldots, n_p$, for which a constraint i_j leaves the active set can be detected by solving

$$0 \approx \lambda_j^{init*} + \frac{d\lambda_j^{init*}}{dp_k}(p_k - p_k^{init*}), \quad \forall j \in A. \tag{9}$$

Likewise, the parameter vector value p_k for which a constraint i_j enters the active set can be detected by solving

$$0 \approx i_j^{init*} + \frac{di_j^{init*}}{dp_k}(p_k - p_k^{init*}), \quad \forall j \in I. \tag{10}$$

The steps of PFA1 are summarized in Algorithm 1:

1. Start from the initial point $(\zeta, \mu, \lambda)^{init^*}$ and compute the function values and derivatives.
2. Compute the boundaries of the sensitivity domain using (9) and (10).
3. Determine the smallest $\Delta p_k^{smallest} = p_k^{smallest} - p_k^{init}$ such that $p_k^{smallest}$ is outside the domain.
4. Recompute the derivative (6) at the domain boundary. This is possible because directional derivatives exist at the domain boundary [26].
5. Compute SBU (7) for $\Delta p_k^{smallest}$, i.e., up to the boundary of the sensitivity domain, and compute SBU (7) with recomputed derivatives from the boundary up until p.

A prerequisite for the algorithm is that conditions C1 to C3 hold and that the starting point fulfills condition C4. Though the authors mention only one change of the active set, the algorithm can be extended to handle several changes. A similar algorithm has been suggested by Ferreau et al. [27] for linear systems.

Another path-following algorithm denoted by PFA2 is formulated by Jäschke et al. [28], where a parametric QP based on the work of Shapiro [23] is solved in each step:

$$\begin{aligned} \min_{\Delta\zeta} \quad & 0.5\Delta\zeta^T L_{\zeta\zeta}^{init*}\Delta\zeta + \Delta\zeta^T L_{\zeta p}^{init*}\Delta p, \\ s.t. \quad & e_\zeta^{init*}\Delta\zeta + e_p^{init*}\Delta p = 0, \\ & i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p = 0, \quad \forall j \in \tilde{A}, \\ & i_j^{init*} + i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p \le 0, \quad \forall j \in \tilde{I}, \end{aligned} \tag{11}$$

with $\tilde{A} = \{j|\lambda_j^{init*} > 0\}$ and $\tilde{I} = \{j|\lambda_j^{init*} = 0\}$. Though this parametric QP is formulated for the more general MFCQ, for comparability, we assume that conditions C1 to C3 hold. In this case, the Lagrange multipliers of (11) are unique and can be directly computed from (11). If $\lambda_j < 0$, $j \in \tilde{A}$, an active constraint becomes inactive. In this case, the authors suggest to reduce Δp as part of the path-following strategy. Furthermore, Δp is also reduced, if (11) is infeasible. The steps of PFA2 are summarized in Algorithm 2:

```
1. Start from initial point $(\zeta, \mu, \lambda)^{init^*}$ and compute function values and derivatives.
2. Set $\Delta p := p - p^{init}$ and $\Delta p^{total} := 0$.
3. While $\Delta p^{total} < p - p^{init}$
   (a) Solve (11) for $\Delta p$.
   (b) If (11) infeasible, set $\Delta p := \alpha_1 \Delta p$ with $\alpha_1 > 0$.
   (c) Elseif $\lambda_j < 0, j \in \tilde{A}$, set $\Delta p := \alpha_2 \Delta p$ with $\alpha_2 > 0$.
   (d) Else, set $\Delta p^{total} := \Delta p^{total} + \Delta p$ and $p^{init} := p^{init} + \Delta p$. For the new initial point, compute the function values related to the constraints and the first- and second-order derivatives. Set $\Delta p := p - p^{init}$.
4. end while.
```

Finally, Yang and Biegler [29] suggested to use the following parametric QP

$$\begin{aligned} \min_{\Delta\zeta} \quad & 0.5\Delta\zeta^T L_{\zeta\zeta}^{init*}\Delta\zeta + \Delta\zeta^T L_{\zeta p}^{init*}\Delta p, \\ s.t. \quad & e_\zeta^{init*}\Delta\zeta + e_p^{init*}\Delta p = 0, \\ & i^{init*} + i_\zeta^{init*}\Delta\zeta + i_p^{init*}\Delta p \preceq 0, \end{aligned} \tag{12}$$

to handle changes of the active set. According to the authors, this is equivalent to the fix-relax strategy (FRS) suggested by Zavala [30].

3.2.2.2. Methods improving a suboptimal initial point. To improve a suboptimal initial point $(\zeta, \mu, \lambda)^{init}$ in a neighborhood of a continuous function ζ^* of local minimizers and to handle changes of the active set, Kadam and Marquardt [31] suggested the following parametric QP

$$\begin{aligned} \min_{\Delta\zeta} \quad & \Phi_\zeta^{init}\Delta\zeta + 0.5\Delta\zeta^T L_{\zeta\zeta}^{init}\Delta\zeta + \Delta\zeta^T L_{\zeta p}^{init}\Delta p, \\ s.t. \quad & e^{init} + e_\zeta^{init}\Delta\zeta + e_p^{init}\Delta p = 0, \\ & i^{init} + i_\zeta^{init}\Delta\zeta + i_p^{init}\Delta p \preceq 0, \end{aligned} \tag{13}$$

which computes a so-called neighboring-extremal update (NEU). According to Guddat et al. [32], the following assumptions have to hold:

A1. A continuous function $\zeta^* : [p_{\min}, p_{\max}] \subset \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$ of local minimizers exists, where $p^{init}, p \in [p_{min}, p_{max}]$.
A2. Conditions C1 and C2 have to hold.
A3. The reduced Hessian has to be positive definite in a neighborhood of ζ^*.
A4. There is a finite number of singular points, for which condition C4 is violated.

The parametric QP (13) is based on the work of Ganesh and Biegler [33]. According to these authors, directional derivatives are not correct if a change in the active set occurs because the Lagrangian is non-differentiable in ζ at those points and, thus, the Hessian would have to be recomputed. Strategies to improve the active set of constraints are suggested by Ganesh and Biegler [33] as well as by Kadam and Marquardt [31].

3.3. *Comparison of fast update methods*

In this section, we will compare and analyze most of the update methods reviewed in Sections 3.1 and 3.2, i.e., QP, RTI, SBU, PFA1, PFA2, FRS and NEU. The update method formulated by Bigelow and Shapiro [22] is not considered independently because it is equivalent to the SBU apart from handling a change in the active set which rarely occurs in practice. For simplicity, we assume that all QPs applied by the fast update methods are feasible.

First, we will compare all fast update methods based on a (parametric) QP with the help of the corresponding KKT conditions listed below:

• KKT conditions of QP (4):

$$\begin{aligned} & \Phi_\zeta^{init,T} + L_{\zeta\zeta}^{init}\Delta\zeta + e_\zeta^{init,T}\mu^{QP} + e_\zeta^{init,T}\lambda^{QP} = 0, \\ & e^{init} + e_\zeta^{init}\Delta\zeta = 0, \\ & i^{init} + i_\zeta^{init}\Delta\zeta \preceq 0, \\ & \lambda^{QP,T}(i^{init} + i_\zeta^{init}\Delta\zeta) = 0, \\ & -\lambda^{QP} \preceq 0. \end{aligned} \tag{4′}$$

• KKT conditions of RTI (5) and NEU (13):

$$\begin{aligned} & \Phi_\zeta^{init,T} + L_{\zeta\zeta}^{init}\Delta\zeta + L_{\zeta p}^{init}\Delta p + e_\zeta^{init,T}\mu^{RTI} + i_\zeta^{init,T}\lambda^{RTI} = 0, \\ & e^{init} + e_\zeta^{init}\Delta\zeta + e_p^{init}\Delta p = 0, \\ & i^{init} + i_\zeta^{init}\Delta\zeta + i_p^{init}\Delta p \preceq 0, \\ & \lambda^{RTI,T}(i^{init} + i_\zeta^{init}\Delta\zeta + i_p^{init}\Delta p) = 0, \\ & -\lambda^{RTI} \preceq 0. \end{aligned} \tag{5′}$$

• KKT conditions of PFA2 (11):

$$\begin{aligned} & L_{\zeta\zeta}^{init*}\Delta\zeta + L_{\zeta p}^{init*}\Delta p + e_\zeta^{init*,T}\mu^{PFA2} + i_\zeta^{init*,T}\lambda^{PFA2} = 0, \\ & e_\zeta^{init*}\Delta\zeta + e_p^{init*}\Delta p = 0, \\ & i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p = 0, \quad j \in \tilde{A}, \\ & i_j^{init*} + i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p \le 0, \quad j \in \tilde{I}, \\ & \lambda^{PFA2,T}(i_j^{init*} + i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p) = 0, \quad j \in \tilde{I}, \\ & -\lambda_j^{PFA2} \le 0, \quad j \in \tilde{I}. \end{aligned} \tag{11′}$$

• KKT conditions of FRS (12):

$$\begin{aligned} & L_{\zeta\zeta}^{init*}\Delta\zeta + L_{\zeta p}^{init*}\Delta p + e_\zeta^{init*,T}\mu^{FRS} + i_\zeta^{init*,T}\lambda^{FRS} = 0, \\ & e_\zeta^{init*}\Delta\zeta + e_p^{init*}\Delta p = 0, \\ & i^{init*} + i_\zeta^{init*}\Delta\zeta + i_p^{init*}\Delta p \preceq 0, \\ & \lambda^{FRS,T}(i^{init*} + i_\zeta^{init*}\Delta\zeta + i_p^{init*}\Delta p) = 0, \\ & -\lambda^{FRS} \preceq 0. \end{aligned} \tag{12′}$$

If we compare the KKT conditions of the suboptimal update methods, QP and RTI, it becomes apparent that they generally provide different updates for $\Delta p \neq 0$. However, if $\Delta p = 0$, both methods provide the same update improving a suboptimal initial point with $\mu^{QP} = \mu^{RTI}$ and $\lambda^{QP} = \lambda^{RTI}$. In addition, it turns out that the KKT conditions of RTI and NEU are always the same [17]. Consequently, both methods provide the same update.

To compare RTI to the remaining QP-based fast update methods, PFA2 and FRS, the residual errors of the KKT conditions, $\mathcal{E}_L$ and $\mathcal{E}_E$, and the deviation from the inequality constraint, $\mathcal{E}_I$, are introduced as in the work of Wolf et al. [34]:

$$\begin{aligned} & \mathcal{E}_L = \Phi_\zeta^{init,T} + e_\zeta^{init,T}\mu^{init} + i_\zeta^{init,T}\lambda^{init}, \\ & \mathcal{E}_E = e^{init}, \\ & \mathcal{E}_I = i^{init}. \end{aligned}$$

After inserting these equations into (5′) and setting $\mu^{RTI} - \mu^{init} = \Delta\mu$ and $\lambda^{RTI} - \lambda^{init} = \Delta\lambda$, we obtain the following equivalent KKT conditions for (5′):

$$\begin{aligned} & \mathcal{E}_L + L_{\zeta\zeta}^{init}\Delta\zeta + L_{\zeta p}^{init}\Delta p + e_\zeta^{init,T}\Delta\mu + i_\zeta^{init,T}\Delta\lambda = 0, \\ & \mathcal{E}_E + e_\zeta^{init}\Delta\zeta + e_p^{init}\Delta p = 0, \\ & \mathcal{E}_I + i_\zeta^{init}\Delta\zeta + i_p^{init}\Delta p \preceq 0, \\ & (\Delta\lambda + \lambda^{init})^T(\mathcal{E}_I + i_\zeta^{init}\Delta\zeta + i_p^{init}\Delta p) = 0, \\ & -(\Delta\lambda + \lambda^{init}) \preceq 0. \end{aligned} \tag{14}$$

It becomes apparent that the first four equations of (14) contain the residual errors and the deviation from the inequality constraints when starting from a suboptimal initial point. Furthermore, the Lagrange multipliers μ^{RTI} and λ^{RTI} correspond to those ones at point (ζ, μ, λ), because $\Delta\mu = \mu - \mu^{init} = \mu^{RTI} - \mu^{init}$ and $\Delta\lambda = \lambda - \lambda^{init} = \lambda^{RTI} - \lambda^{init}$. For an optimal initial point with $\mathcal{E}_L = \mathcal{E}_E = 0$ and $\mathcal{E}_I \le 0$, (14) reduces to

$$\begin{aligned} & L_{\zeta\zeta}^{init*}\Delta\zeta + L_{\zeta p}^{init*}\Delta p + e_\zeta^{init*,T}\Delta\mu + i_\zeta^{init*,T}\Delta\lambda = 0, \\ & e_\zeta^{init*}\Delta\zeta + e_p^{init*}\Delta p = 0, \\ & i^{init*} + i_\zeta^{init*}\Delta\zeta + i_p^{init*}\Delta p \preceq 0, \\ & (\Delta\lambda + \lambda^{init*})^T(i^{init*} + i_\zeta^{init*}\Delta\zeta + i_p^{init*}\Delta p) = 0, \\ & -(\Delta\lambda + \lambda^{init*}) \preceq 0. \end{aligned} \tag{15}$$

For (11′), we set $\mu^{PFA2} = \Delta\mu$ and $\lambda^{PFA2} = \Delta\lambda$ and, with $\Delta\lambda_j = \Delta\lambda_j + \lambda_j^{init*}$ for $j \in \tilde{I}$, we obtain

$$\begin{aligned}
&L_{\zeta\zeta}^{init*}\Delta\zeta + L_{\zeta p}^{init*}\Delta p + e_{\zeta}^{init*,T}\Delta\mu + i_{\zeta}^{init*,T}\Delta\lambda = 0,\\
&e_{\zeta}^{init*}\Delta\zeta + e_{p}^{init*}\Delta p = 0,\\
&i_j^{init*} + i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p \le 0, \quad j \in \tilde{I},\\
&(\Delta\lambda_j + \lambda_j^{init*})^T(i_j^{init*} + i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p) = 0, \quad j \in \tilde{I},\\
&-(\Delta\lambda_j + \lambda_j^{init*}) \le 0, \quad j \in \tilde{I},\\
&i_{j,\zeta}^{init*}\Delta\zeta + i_{j,p}^{init*}\Delta p = 0, \quad j \in \tilde{A},
\end{aligned} \tag{16}$$

The KKT systems of (15) and (16) are equivalent as long as no strongly active constraint becomes inactive, because the first two equations are the same and the third to fifth equations are equivalent for i_j with $j \in \tilde{I}$. In this case, RTI and PFA2 provide the same update.

For (12′), we set $\mu^{FRS} = \Delta\mu$ and $\lambda^{FRS} = \Delta\lambda$. We obtain

$$\begin{aligned}
&L_{\zeta\zeta}^{init*}\Delta\zeta + L_{\zeta p}^{init*}\Delta p + e_{\zeta}^{init*T}\Delta\mu + i_{\zeta}^{init*T}\Delta\lambda = 0,\\
&e_{\zeta}^{init*}\Delta\zeta + e_{p}^{init*}\Delta p = 0,\\
&i^{init*} + i_{\zeta}^{init*}\Delta\zeta + i_{p}^{init*}\Delta p \preceq 0,\\
&\Delta\lambda^T(i^{init*} + i_{\zeta}^{init*}\Delta\zeta + i_{p}^{init*}\Delta p) = 0,\\
&-\Delta\lambda \preceq 0,
\end{aligned} \tag{17}$$

The KKT conditions of (15) and (17) are equivalent if $\Delta\lambda = \Delta\lambda + \lambda^{init*}$, i.e., if $A = \emptyset$. In this case, RTI and FRS provide the same update.

After the QP-based fast update methods have been compared with the help of their KKT conditions, all fast update methods will be examined for a constant and a changing active set, respectively.

3.3.1. *Constant active set*

For a constant active set, the update provided by PFA1 always corresponds to SBU, because $p^{smallest} > p$ (see Algorithm 1).

3.3.1.1. *Optimal initial point.*

If we insert (6) into (7) for an optimal initial point, we obtain

$$\begin{aligned}
&L_{\zeta\zeta}^{init*}\Delta\zeta + L_{\zeta p}^{init*}\Delta p + e_{\zeta}^{init*,T}\Delta\mu + i_{\zeta}^{a,init*,T}\Delta\lambda^a = 0,\\
&e_{\zeta}^{init*}\Delta\zeta + e_{p}^{init*}\Delta p = 0,\\
&i_{\zeta}^{a,init*}\Delta\zeta + i_{p}^{a,init*}\Delta p = 0,
\end{aligned} \tag{18}$$

which corresponds to a first-order correction of the KKT conditions (3) assuming a constant active set. By comparison, (18) corresponds to (15) for a constant active set. Consequently, SBU, PFA1, RTI, NEU and PFA2 provide the same update in this case.

3.3.1.2. *Suboptimal initial point.*

If SBU is applied for a suboptimal initial point, i.e., if the derivatives are computed by

$$\begin{bmatrix} \dfrac{d\zeta}{dp} \\ \dfrac{d\mu}{dp} \\ \dfrac{d\lambda^a}{dp} \end{bmatrix}^{init} = -\mathcal{M}^{init,-1}\begin{bmatrix} L_{\zeta p} \\ e_p \\ i_p^a \end{bmatrix}^{init},$$

SBU corresponds to a first-order correction with respect to p of the following system:

$$\begin{aligned}
&\Phi_{\zeta}^T + e_{\zeta}^T\mu + i_{\zeta}^{a,T}\lambda^a - \mathcal{E}_L = 0,\\
&e - \mathcal{E}_E = 0,\\
&i^a - \mathcal{E}_I^a = 0.
\end{aligned}$$

Consequently, the update provided by SBU sustains the residual errors and the deviation from the active inequality constraints, i.e., SBU does not contain a correction term to improve a suboptimal initial point inherently. If $\Delta p = 0$, it follows that $\Delta\zeta = 0$, which also applies for the updates provided by PFA1, PFA2 and FRS. For a suboptimal initial point, RTI reduces to the so-called approximate tangential predictor (cf. [14])

$$\begin{bmatrix} \zeta \\ \mu \\ \lambda^a \end{bmatrix} = \begin{bmatrix} \zeta \\ \mu \\ \lambda^a \end{bmatrix}^{init} - \mathcal{M}^{init,-1}\left(\begin{bmatrix} L_{\zeta p} \\ e_p \\ i_p^a \end{bmatrix}^{init}\Delta p + \begin{bmatrix} \mathcal{E}_L \\ \mathcal{E}_E \\ \mathcal{E}_I^a \end{bmatrix}^{init}\right), \tag{19}$$

which consists of a prediction term and a correction term based on the size of the residual errors as well as the deviation from the active inequality constraints and which can be directly derived from (14). To summarize, SBU, PFA1 and PFA2 provide the same update. This update differs from the update provided by RTI and NEU which contains a correction term for a suboptimal initial point.

3.3.2. *Changing active set*

SBU is the only fast update method which does not account for changes in the active set. Thus, the update of SBU only corresponds to the one of PFA1 until $p^{smallest}$ is reached. Then, PFA1 recomputes the derivative such that a piecewise linear function $F : \mathbb{R}^{n_p} \to \mathbb{R}^{n_\zeta}$ is obtained which comprises a discontinuity at every point where the active set changes.

3.3.2.1. *Optimal initial point.*

Though the residual errors and the deviation from the active inequality constraints vanish for an optimal initial point, PFA1 and RTI (and thus also NEU) only coincide until $p^{smallest}$ is reached. This is due to the fact that the Hessian of the Lagrangian is not updated for RTI (cf. Section 3.2.2.2). If no active constraint becomes inactive, PFA2, RTI and NEU provide the same update. Otherwise, the update provided by PFA2 will be computed in a path-following manner. Hence, in this case only RTI and NEU always provide the same update.

3.3.2.2. *Suboptimal initial point.*

For a change in the active set and a suboptimal starting point, only RTI and NEU provide the same update.

3.3.3. *Summary*

Fig. 1 summarizes the results of the comparison of the fast update methods which provide a fast but typically suboptimal update of the control variables based on the equations presented in Sections 3.1 and 3.2. Note that if two methods provide the same update only in special cases, such as RTI and FRS if $A = \emptyset$, these methods are not connected by a line. The similarities between the various methods are indeed surprising and constitute a major contribution of this work. It should be noted, that the authors of the works presented in this section have typically not attempted to analyze the relations between their suggested new method and existing methods. Exceptions are the contributions of Diehl et al. [14] and Würth et al. [17].

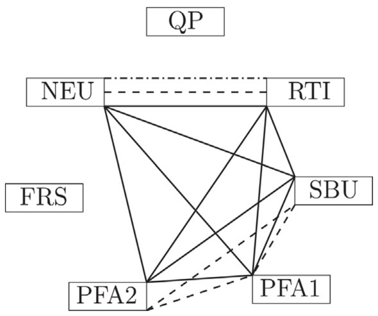

Fig. 1. Comparison of fast update methods under parametric changes. Lines connect methods which provide the same update. Solid line: in case of a constant active set and optimal initial point. Dashed line: in case of a constant active set and suboptimal initial point. Dashed dotted line: in case of a changing active set.

3.4. Illustrative example

In order to illustrate the results of Section 3.3, the following PNLP is considered:

$$\begin{aligned}\min_{\zeta}\quad & (\zeta_1)^2+(\zeta_2)^2+2p(\zeta_1+\zeta_2)+\zeta_2,\\ s.t.\quad & -\zeta_1+p\le 0,\\ & 2(\zeta_1)^2+\zeta_2-10\le 0,\\ & -\zeta_2+0.5\le 0.\end{aligned}$$

We compare the updates of the seven update methods QP, RTI, SBU, PFA1, PFA2, FRS and NEU. For PFA2, we set $\alpha_2 := 0.8$ and terminate the iterations after (11) has been conducted 100 times. For each fast update method, we will consider three different cases:

1. The initial point is optimal and the active set does not change.
2. The initial point is again optimal but the active set changes.
3. The initial point is suboptimal and the active set changes.

For each of these cases and each of the fast update methods, several updates are computed for different values of *p* while the updates are all based on the same initial point and the same initial parameter vector specified for the individual scenario.

3.4.1. Optimal initial point and constant active set

The optimal initial point is given by $\zeta^{init*}=[1.95,\ 2.366]^T$ and $\lambda^{init*}=[0,\ 0.2678,\ 0]^T$ for $p^{init}=-3$ as shown in Fig. 2a. The parameters *p* for which updates are computed based on p^{init} are −3, −2.8, −2.6, −2.4 and −2.2. For $p=p^{init}$, we see that all updates correspond to the optimal solution. For $p>p^{init}$, the update provided by QP exactly corresponds to the initial optimal solution because changes in the parameter *p* are not considered by QP (4). As a result, the update provided by QP deviates from the updates provided by all other methods. As shown in Fig. 1, the fast update methods RTI, SBU, PFA1, PFA2 and NEU compute the same update. Since no change of the active set occurs, SBU and PFA1 are the same. As mentioned above, the update computed by SBU and RTI are the same for a constant active set when starting from an optimal initial point. Furthermore, the updates computed by PFA2 and RTI are the same because no active constraint becomes inactive. Finally, RTI and NEU are the same due to their corresponding KKT conditions. The update provided by FRS does not correspond to the one of RTI, because $A\neq\emptyset$.

3.4.2. Optimal initial point and changing active set

The optimal initial point for $p^{init}=-1.6$ is given by $\zeta^{init*}=[2.040,\ 1.68]^T$ and $\lambda^{init*}=[0,\ 0.00393,\ 0]^T$ as shown in Fig. 2b. The parameters *p* explored are −1.6, −1.8, −2.0545, −2.2, −2.4 and −2.6. For $p=p^{init}$, we see that all updates again correspond to the optimal initial point. As described above, the QP update again deviates from the other updates. For a changing active set, the update provided by SBU deviates from the ones provided by PFA1, PFA2 and RTI for $p<-2.0545$, because the active set is not updated by SBU. Since the initial point is optimal and $A=\emptyset$, FRS provides the same update as RTI.

3.4.3. Suboptimal initial point and changing active set

The suboptimal initial point is given by $\zeta^{init}=[2.018,\ 1.85]^T$, $\lambda^{init}=[0,\ 0.00946,\ 0]^T$ for $p^{init}=-2.2$ as shown in Fig. 2c. The parameters *p* investigated are −2.2, −2.0545, −1.88, −1.6 and −1.4. The QP update improves the suboptimal initial point, because the KKT conditions (3) are not fulfilled. For all other parameters *p*, the update of the QP again corresponds to the first update. Both RTI and NEU improve the suboptimal initial point and update the active set. For SBU, the first update corresponds to the suboptimal initial point because SBU does not improve suboptimal initial points as discussed in Section 3.3. This is also true for PFA1, but in contrast to SBU the method detects the change of the active set for $p>-2.0545$. Note that the slope of the tangential predictor of PFA1 does not correspond to the one of RTI, because PFA1 recomputes the Hessian at the point where the active set changes. The updates of PFA2 correspond to the one of SBU and PFA1 for $p<-2.0545$. However, for $p>-2.0545$ the algorithm approaches the point where the active set changes without passing it and the iterations are stopped after the iteration limit is reached. The update of FRS again does not correspond to the one of *RTI* because $A\neq\emptyset$.

3.5. Reduction of computational time for fast update methods

Several approaches have been suggested to reduce the computational time of a fast update, if computational delay is not negligible. Besides, for example, applying efficient approximations for the derivatives and function values related to the equality and inequality constraints [13], QP (4) can be reformulated with the help of an augmented Lagrangian and solved by projected successive over-relaxation [35]. Furthermore, NEU (13) can be solved in a distributed manner as suggested by Wolf et al. [36].

3.6. Relation to nonlinear parametric programming

Nonlinear parametric programming (NLPP) determines the optimal solution to (2) under parametric changes for $p\in\mathbb{R}$ by tracking the KKT conditions of (3). Hence, it is closely related to the sensitivity-based update methods introduced in Section 3.2. Therefore, NLPP is considered in more detail in this section.

Like the sensitivity-based updated methods, NLPP originated from sensitivity analysis (cf. [19,20]). An excellent review on NLPP is given by Seferlis and Hrymak [37]. NLPP can be interpreted as continuation of the solution of optimization problems because continuation methods developed for continuation of the solution of nonlinear equations are employed in NLPP to track the solution to (2) (see, for example, Guddat et al. [32]). However, in contrast to continuation of the solution of nonlinear equations [38], the KKT system is usually not a system of equations but also comprises inequality constraints. To handle inequality constraints and determine the optimal solution curve, methods that employ active index set strategies can be applied among others [32]. These methods will be outlined next assuming A1–A4 (cf. Section 3.2.2).

A point $y=[\zeta^T,\mu^T,\lambda^T)]^T$ with $y\in\mathbb{R}^{n_y}$ that fulfills the KKT conditions (3) for a given parameter value $p^{init}\in\mathbb{R}$ is termed a zero point y^0. If condition C4 also holds for zero point $y^0(p^{init})$, then $y^0(p^{init})$ is a regular point, i.e., the Jacobian of $F(y,p):=[L_\zeta^T,\ e^T,\ i^{a,T}]^T$

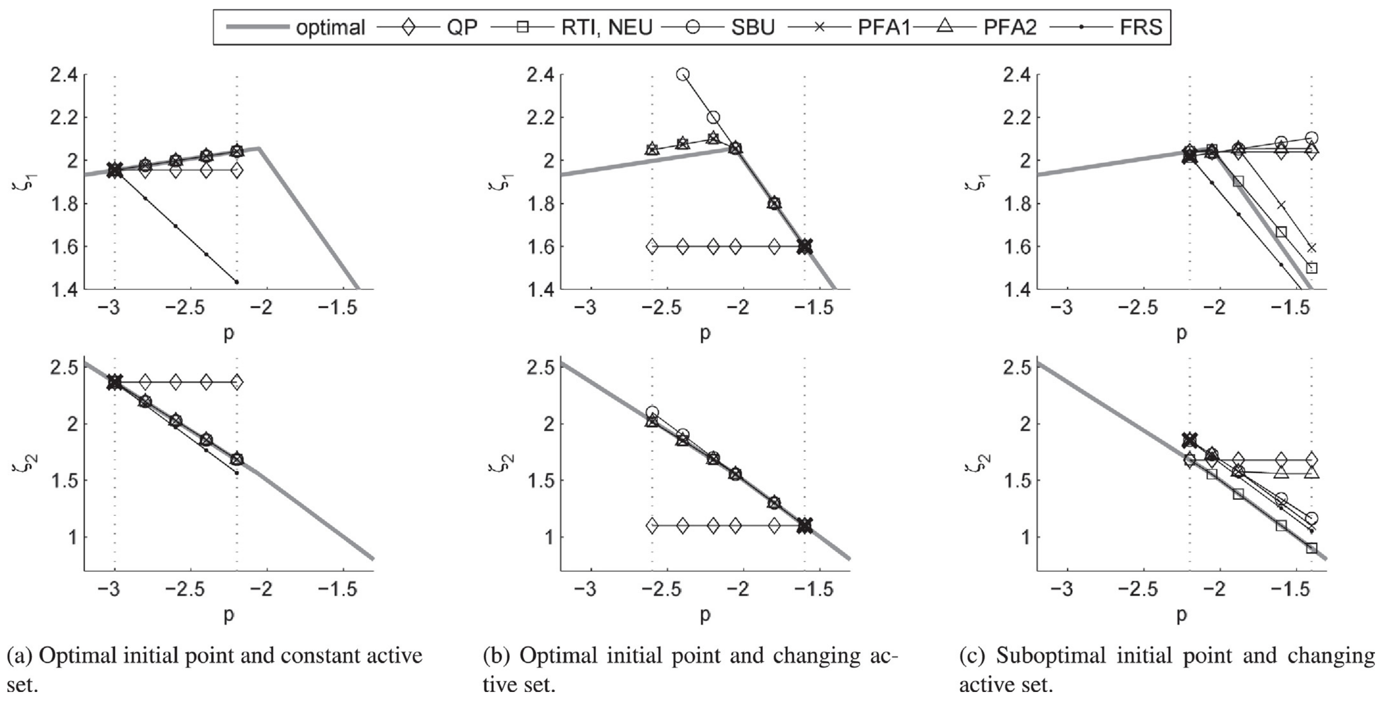

Fig. 2. Performance of suboptimal and sensitivity-based update methods. Each marker represents a fast update of its respective fast update method. The initial point is represented by a bold cross.

at zero point $y^0(p^{init})$ has full rank n_y. For a regular point $y^0(p^{init})$, a smooth function y^0 of zero points with constant active set exists such that $F(y^0(p), p) = 0$ within an open interval containing p^{init} (cf. Section 3.2.1). According to Guddat et al. [32], standard continuation methods such as predictor–corrector (PC) methods can be applied in this interval. Like other continuation methods, PC methods determine the solution to a smooth system of nonlinear equations

$$F(y, p) = 0, \tag{20}$$

with $F : \mathbb{R}^{n_y} \times [p_{\min}, p_{\max}] \to \mathbb{R}^{n_y}$ and $[p_{\min}, p_{\max}] \subset \mathbb{R}$, starting from a zero point y^0 for a given parameter value p^{init} [38]. The predictor consists of an Euler step:

$$\tilde{y}(p_{k+1}) = y^0(p^{init}) + y^0_p(y^0(p^{init}), p^{init})h^{init}, \tag{21}$$

where step direction $y^0_p(y^0(p^{init}), p^{init}) := -F_y^{-1}(y^0(p^{init}), p^{init})F_p(y^0(p^{init}), p^{init})$ is derived from $dF(y(p^{init}), p^{init})/dp = 0$. The step length $h^{init} = p_{k+1} - p^{init}$ is restricted or altered to guarantee closeness to the solution manifold. The corrector consists of a Newton step. Since (20) is under-determined, the direction of the corrector step is a degree of freedom and can be determined in several ways [38]. For $p := p_{k+1}$, we obtain

$$y(p) = \tilde{y}(p) - F_y^{-1}(\tilde{y}(p), p)F(\tilde{y}(p), p). \tag{22}$$

If condition C4 is violated, a singular point is crossed and the active set might change. If the active set changes, at least one inequality constraint becomes weakly active, i.e., one Lagrange multiplier λ_i^a vanishes or one inactive inequality constraint becomes active. If we assume that exactly one inequality constraint becomes weakly active at each singular point, the singular point does not have to be detected by the continuation method, but the active set can be updated a posteriori by checking the Lagrange multipliers λ_i^a and the former inactive inequality constraints [32]. How to handle more general singular points is described by Guddat et al. [32].

To point out the similarity between NLPP and the sensitivity-based update methods introduced in Section 3.2, we will qualitatively compare an update provided by a PC method with one provided by NEU (13) as a representative of the sensitivity-based update methods. The PC method applies (21) as a predictor and (22) as a corrector. For comparison, it is assumed that $h^{init} = \Delta p$. Note that this is usually not the case because h^{init} is adapted to stay close to the solution manifold, while Δp is measured or estimated and thus fixed to a pre-specified value. First, we will consider the case, where the active set stays constant. In this case, the update of NEU reduces to the approximated tangential predictor (19). If we start from a suboptimal initial point $y(p^{init})$ as shown in Fig. 3a, both the PC method and the NEU conduct a predictor as well as a corrector step. However, the NEU uses the derivatives for the predictor and corrector computed at p^{init}, while the PC method recomputes the derivatives for the corrector step. If we start from an optimal initial point $y^0(p^{init})$ as shown in Fig. 3b, we see that the predictor step (21) corresponds to the NEU update. However, the corrector step is solely carried out for the PC method. Thus, we can interpret NEU as a corrector-predictor (CP) method with approximated derivative z_p, for which the preceding corrector step is only conducted if the initial point is suboptimal. In Fig. 3c, a change in the active set is considered starting from an optimal initial point. We see that NEU tracks the change of the active set. No correction is computed because the initial point is optimal. On the other hand, the PC method does not detect the change in the active set until the correction step is carried out.

Summarizing, besides many similarities, the main difference between the fast updates of Section 3.2 and the PC methods is that PC methods determine the step length $h(p_1)$ such that closeness to the solution manifold can be guaranteed. Moreover, only one parameter instead of a parameter vector is considered in the PC methods. Furthermore, the fast sensitivity-based update methods do not conduct a corrector step if the initial point is optimal. Finally, active set changes are often handled by a QP by the fast sensitivity-based update methods (cf. Section 3).

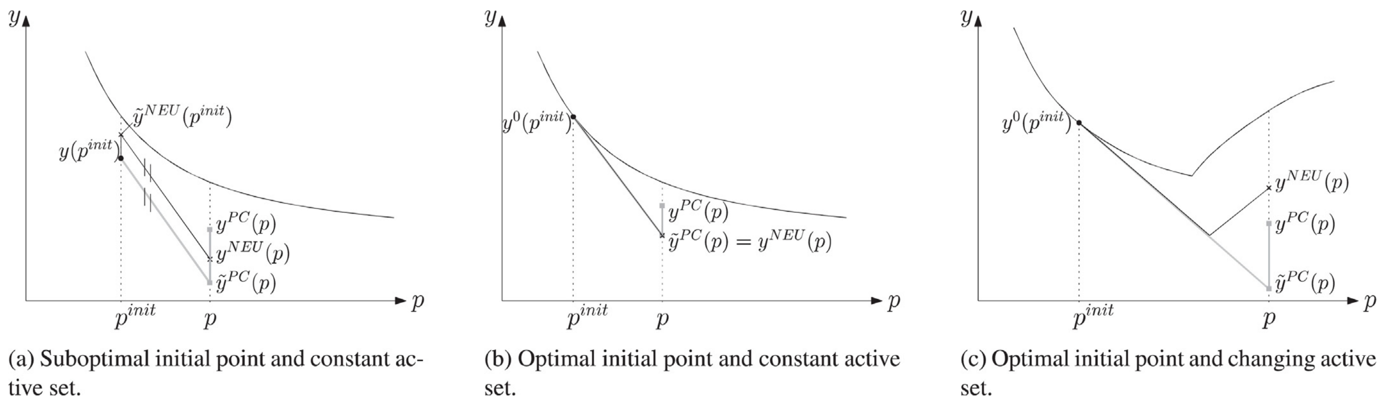

Fig. 3. Comparison of updates provided by the PC method and by the NEU at p. The PC method is denoted by the superscript PC and the NEU by the superscript NEU.

3.7. *Discussion*

It has been shown that the fast update methods for the PNLP (2) handle parametric changes, changes of the active set and suboptimal initial points rather differently. Consequently, the method of choice strongly depends on the PNLP, the number of expected changes in the active set and the initial point. Moreover, the computational delay arising when applying a fast update method also has to be taken into account.

In general, it pays off to apply a fast update method other than QP (4) if p is contained at least linearly in the constraints or bilinearly in the objective function as in the illustrative example above. Though SBU is a very fast update method as only a system of linear equations has to be solved, active set changes remain undetected during the update and suboptimal initial points are not improved. The advantage of PFA1 is that active set changes are detected and that the path-following method can be stopped if too many changes of the active set occur. Nonetheless, computational delay might still be critical, because function values related to the constraints as well as first- and second-order derivatives are recomputed for every change in the active set. Furthermore, PFA1 does not improve a suboptimal initial point. In contrast to all other methods, PFA2 covers the case that (11) might be infeasible if Δp becomes too large by subdividing Δp into smaller intervals in a path-following manner. However, the path-following strategy suggested for the case when an active constraint becomes inactive may lead to an infinite number of iterations. A solution to this problem might be to determine the point, where the active set changes similar to PFA1. Another drawback of PFA2 is that computational delay might not be negligible if (11) is solved too often. Again, PFA2 does not improve a suboptimal initial point. Both RTI and NEU detect changes of the active set and inherently provide a correction term when starting from a suboptimal initial point. Furthermore, they reduce to other fast update methods for an optimal initial point and a constant active set (cf. Fig. 1). Though RTI and NEU have been developed independently, their KKT conditions coincide. Thus, they bridge the gap between suboptimal and sensitivity-based update methods. The computational time can nowadays be reduced to the range of microseconds for smaller scale models (cf. [39] for RTI), if algorithmic improvements and efficient approximations for the derivatives are applied.

4. Fast NMPC schemes tailored to a single-layer control architecture

In this section, a review will be given on fast NMPC schemes that apply one of the fast update methods introduced in Section 3 and that are tailored to a single-layer control architecture.

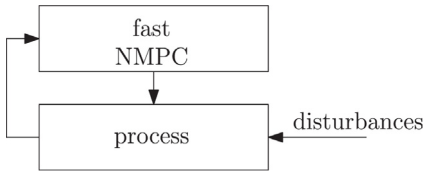

Fig. 4. Single-layer control architecture for fast NMPC schemes.

In literature, different classes of NMPC are used depending on the underlying system considered. For discrete-time systems, discrete-time NMPC is applied, where the sampling period must be correlated to the fastest time constant of the system and the controls must be chosen as a multiple of the equidistant sampling period Δt. For continuous-time systems, sampled-data NMPC is typically applied if the computational delay to solve PNLP (2) to convergence is not negligible. Here, the continuous-time system is subject to sampled state information, i.e., the current state is sent to the controller at each sampling instant and the fast update of the control variables is applied in open loop until the next sampling instant (cf. [40]). Though the states, the controls, the constraints and the disturbances can be discretized independently from the sampling period for this NMPC class (cf. Section 2.2), the control variables are often discretized to be a multiple of the sampling period. This specific class of sampled-data NMPC will be termed sample-and-hold NMPC in the following. For brevity, the closed-loop formulations of discrete-time NMPC and sampled-data NMPC will not be stated in detail. However, the remainder of this section is presented such that the results can be transferred to both NMPC classes.

As shown in Fig. 4, a single-layer control architecture for fast NMPC schemes consists of the process layer as well as of a centralized fast NMPC. At each sampling instant, the fast NMPC has to provide a fast update to (2) applying one of the methods described in Section 3. Here, the fast update approximates the solution to (1) on the receding time horizon $\mathcal{I}_s := [t_{0,s}, t_{f,s}]$, where index s denotes all quantities related to $\mathcal{I}_s$. While the start time $t_{0,s}$ always coincides with the current sampling instant, which will also be denoted by $t_{0,s}$, the final time $t_{f,s}$ is determined by the horizon setting. If a moving horizon setting is applied, the horizon length $t_{f,s} - t_{0,s}$ is fixed to a pre-specified value for each time horizon $\mathcal{I}_s$. If a shrinking horizon setting is chosen, the horizon length is typically reduced by Δt when moving from $\mathcal{I}_s$ to $\mathcal{I}_{s+1}$. Note, however, that the special case of an optimized final time is closely related to the shrinking horizon setting and hence covered like this. For the fast update to be computed, the current parameter vector p_s is first measured or estimated at each sampling instant $t_{0,s}$. If available, new disturbance predictions are also included in p_s. After the fast update has been

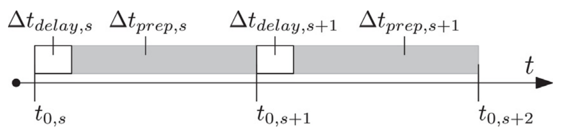

Fig. 5. Operational phases of the fast NMPC. White box: time interval occupied by the feedback phase. Gray box: time interval occupied by the online preparation phase.

determined for p_s, the corresponding control trajectories u_s are sent to the process with negligible computational delay and are applied between sampling instants $t_{0,s}$ and $t_{0,s+1}$ before new measurements are taken.

In the following, the offline preparations and the operational phases of the fast NMPC are described in detail. Here, we will address when and how the initial point, the derivatives and the function values related to the constraints are computed by the fast NMPC so that one of the methods described in Section 3 can directly be applied as soon as measurements become available. In the review of Diehl et al. [14], the offline preparations and the operational phases have already been listed together with other strategies reducing or accounting for computational delay such as the shift initialization strategy and delay compensation by prediction. In contrast to Diehl et al. [14], further strategies are identified in this section and arranged in such a way that a direct comparison and assessment of fast NMPC schemes become possible. Moreover, the assessment is supported by a case study which is based on the results of this section as well as the results of Section 3.

4.1. Offline preparations

As noted by Diehl et al. [14], code improvements related to model evaluations, which can significantly reduce computational effort, can be done offline. For example, automatic differentiation routines can be applied to the model subroutines such that fast first- and second-order derivatives can be provided online (see, for example, the works of Houska et al. [39] and of Hannemann et al. [41]). Besides code improvements, expensive computations are carried out offline in order to reduce computational delay. For instance, the nominal optimal solution $(\zeta, \mu, \lambda)^{nom}$, the corresponding function values related to the equality and inequality constraints and the corresponding derivatives are computed for a nominal parameter vector p^{nom}. Here, p^{nom} is usually chosen such that it represents a steady state on which the plant is operated. Another alternative is that it is determined by integrating forward. Typically, the nominal solution as well as the function values and derivatives are only applied for the first update on horizon $\mathcal{I}_1$. For some cases, however, the nominal solution is reused at every sampling instant. Note that we approach explicit MPC methods if the nominal optimal solution is computed offline for all admissible p^{nom}. This case will be discussed in more detail in Section 6.3.

4.2. Operational phases of centralized controller

In general, two operational phases can be distinguished as part of the strategies applied to reduce computational delay: the online preparation phase and the feedback phase [42,43]. The two operational phases of the fast NMPC are depicted in Fig. 5. In the online preparation phase, the available time $\Delta t_{prep,s}$ between two sampling instants, $t_{0,s}$ and $t_{0,s+1}$, is exploited to conduct as many precomputations as possible that are required in the successive feedback phase. In the feedback phase, a fast update $(\zeta, \mu, \lambda)_{s+1}$ is calculated based on the current p_{s+1} using all information available from the online preparation phase. The fast update is then immediately sent to the process. As a consequence, the computational delay of the feedback phase $\Delta t_{\text{delay},s+1}$ is often negligible. In addition, further computations might be carried out in the feedback phase which, however, lead to an increase in computational delay.

4.2.1. Online preparation phase

In the time interval between sampling instants $t_{0,s}$ and $t_{0,s+1}$, computations are made for the next feedback phase that have to be completed before new measurements become available at sampling instant $t_{0,s+1}$. If the computational delay of the previous feedback phase was negligible, the available time $\Delta t_{prep,s}$ of the online preparation phase coincides with the sampling period Δt. However, if the computational delay of the previous feedback phase was not negligible, the time available for computations has to be reduced by the computational time $\Delta t_{\text{delay},s}$ required to receive update $(\zeta, \mu, \lambda)_s$.

The computations performed in the online preparation phase depend on the choice of the reference parameter vector p^{ref} and the reference point $(\zeta, \mu, \lambda)^{ref}$. Here, two cases can be distinguished:

- The reference corresponds to the *nominal solution*, i.e., $p^{ref} := p^{nom}$ and $(\zeta, \mu, \lambda)^{ref} := (\zeta, \mu, \lambda)^{nom}$.
- The reference corresponds to the *previous solution* of horizon $\mathcal{I}_s$, i.e., $p^{ref} := p_s$ and $(\zeta, \mu, \lambda)^{ref} := (\zeta, \mu, \lambda)_s$.

Based on the choice of the reference, the initial parameter vector p^{init}_{s+1} and the initial point $(\zeta, \mu, \lambda)^{init}_{s+1}$ are computed for horizon $\mathcal{I}_{s+1}$ applying one of four initialization strategies:

- If the *direct initialization strategy* (DIS) is applied (cf., for example, [44]), $p^{init}_{s+1} := p^{ref}$ and $(\zeta, \mu, \lambda)^{init}_{s+1} := (\zeta, \mu, \lambda)^{ref}$. DIS is based on the assumption that the parameter vector p_{s+1} just differs slightly from p^{ref}. Since the horizon length stays the same as ζ^{init}_{s+1} contains the control parameters from the previous horizon, this initialization strategy can be only applied in a moving horizon setting.
- The *shift initialization strategy* (SIS) is based on Bellman's principle of optimality [45], which states that the remaining decisions of an optimal policy again constitute an optimal policy with respect to the state that results from the first decisions in the absence of disturbances. Thus, $p^{init}_{s+1} := x^{ref}(t_{0,s+1})$ and $(\zeta, \mu, \lambda)^{init}_{s+1} := (\zeta, \mu, \lambda)^{ref,red}$. Here, $(\zeta, \mu, \lambda)^{ref,red}$ contains all elements of $(\zeta, \mu, \lambda)^{ref}$, which are not outdated for time interval $[t_{0,s+1}, t_{f,s}]$. If the nominal solution is the reference, all information required to construct p^{init}_{s+1} and $(\zeta, \mu, \lambda)^{init}_{s+1}$ is already available. Otherwise, it can be computed by a forward integration. SIS is specifically tailored to an optimal reference in a shrinking horizon setting. However, it can also be applied if the reference is suboptimal. Furthermore, it can be extended to a moving horizon setting by prolonging the horizon (cf. [14,46,47]). Note that the reference cannot be based on the nominal solution if $t_{0,s+1} > t^{nom}_f$.
- The *optimal initialization strategy* (OIS) has been introduced by Zavala and Biegler [21]. It improves a suboptimal initial point, which may be provided by one of the other two initialization strategies, by optimizing it to convergence. A prerequisite is, however, that the optimal solution and the corresponding function values and derivatives can be determined within the time interval available for the online preparation phase.
- If the optimal solution cannot be determined in the time interval available for the online preparation phase, we propose the *iterative initialization strategy* (IIS). Here, as many iterations as possible are conducted to improve the initial points provided by SIS and DIS, respectively.

Fig. 6 summarizes possible choices of the reference and the initialization strategy for a moving and shrinking horizon setting, respectively. If the nominal solution is taken as a reference in a

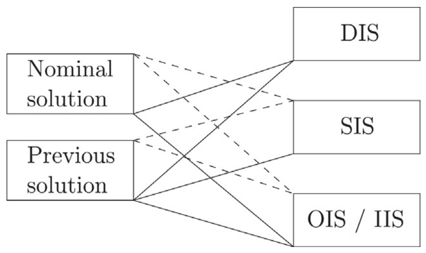

Fig. 6. Choices of reference and initialization strategy. Lines connect possible choices for reference and initialization strategy. Solid line: moving horizon setting. Dashed line: shrinking horizon setting.

moving horizon setting, all possible initialization strategies (DIS, OIS and IIS) provide the optimal solution because the reference point $(\zeta, \mu, \lambda)^{ref} := (\zeta, \mu, \lambda)^{nom}$ is already optimal for $p^{ref} := p^{nom}$. Likewise, if the nominal solution is taken as a reference in a shrinking horizon setting, all possible initialization strategies (SIS, OIS and IIS) again provide the optimal solution. Note that these initialization strategies can also be applied to receive a good initial guess ζ_{s+1}^{init} if PNLP (2) is solved by an iterative solution strategy at each sampling instant.

If the nominal solution is chosen as a reference in a moving horizon setting, the optimal function values related to the constraints and the derivatives correspond to the nominal ones. If the nominal solution is chosen as a reference in a shrinking horizon setting, these values do not have to be computed but can be assembled from the nominal solution, because Bellman's principle of optimality applies. However, if the previous solution is chosen as a reference, the function values and the derivatives must be recomputed for the feedback phase of horizon $\mathcal{I}_{s+1}$. In general, four strategies can be found in the literature (see, for example, Bock et al. [13]):

- All function values and derivatives are recomputed.
- The function values and derivatives are recomputed except for the Hessian which is approximated.
- The function values and the first-order derivative of the objective function are recomputed. The remaining derivatives are approximated.
- The function values are recomputed and the derivatives are approximated.

4.2.2. Feedback phase

In the feedback phase, a fast update is computed for the current parameter vector p_{s+1} based on p_{s+1}^{init} and $(\zeta, \mu, \lambda)_{s+1}^{init}$ by one of the methods described in Section 3. In contrast to iterative solution procedures, which solve PNLP (2) to convergence for p_{s+1} starting at ζ_{s+1}^{init}, the fast update is usually computed with negligible computational delay as most of the computations have already been performed in the preparation phases. However, if more computations are conducted in the feedback phase, computational delay may arise. This particular case will be termed delayed feedback in the following. Delayed feedback might be encountered, for example,

- if computations are carried out after measurement values become available such as a linearization of the model around the current state,
- if the active set is readjusted after the update is computed (see, for example, Yang and Biegler [29] for a strategy with clipping in the first interval) and
- if additional QP iterations are conducted.

If possible, it is advantageous to send the delayed feedback immediately to the process to apply it during the remainder of the sampling interval. If this is not possible, e.g. due to communication protocols, the information has to be implemented at the next sampling instant. In both cases, delay compensation by prediction, i.e., delay compensation by simulating the model forward in time, can be applied to improve the performance of the controller.

4.3. Fast NMPC schemes in literature

All of the reported fast NMPC schemes apply in some combination the elements for performance tuning presented in the previous subsections as well as one of the fast update methods introduced in Section 3. The major lines of developments are introduced in the following emphasizing the specific elements used to contribute to a fast response of the fast NMPC.

4.3.1. Real-time optimization by Bock et al. [42]

The first fast sample-and-hold NMPC scheme has been suggested by Bock et al. [42] already in 2000. However, no stability proof has been given. In the online preparation phase, the previous solution is taken as a reference and SIS is applied. In the feedback phase, QP iterations are conducted until convergence is reached. Delay compensation by prediction is used.

4.3.2. Real-time iterations by Diehl et al. [18]

The first fast NMPC scheme, that conducts as many calculations as possible in the online preparation phase *and* applies one of the fast update methods introduced in Section 3, has been based on RTI. It is a fast sample-and-hold NMPC scheme initially introduced by Diehl et al. [18] without stability considerations. In the online preparation phase, the previous solution is taken as a reference and SIS is applied. Furthermore, the first-order derivatives and the functions values related with the constraints are computed and the Hessian is approximated. In the feedback phase, RTI (5) is conducted. Four variants of the real-time iterations, which speed up the computation of the fast update and which differ in the online preparation phase, have been introduced by Bock et al. [13] (cf. Section 5.1). Furthermore, a lot of algorithmic improvements have been introduced in recent years such as partially reduced SQP methods by Schäfer et al. [48], adjoint sensitivity generation by Wirsching et al. [49] and by Kirches et al. [50] and autogenerated C-code by Houska et al. [39] (cf. also the review by Kirches et al. [51]). Convergence has been shown by Diehl et al. [52,53] and a nominal stability proof for a problem formulation without inequality constraints has been presented by Diehl et al. [52]. Over the years, real-time iterations have been applied to control robots [54], kites [55,53,56], aircrafts [57], combined cycle power plants [58], gasoline engines [59] and distillation columns [48].

4.3.3. Neighboring-extremal controller by Würth et al. [17]

Based on the work of Kadam and Marquardt [31] (see Section 5.1), the authors introduce a fast sample-and-hold economic NMPC scheme without stability proof. In the online preparation phase, the previous solution is taken as a reference and SIS is applied. The function values and derivatives are assembled from computations of the previous feedback phase. In the feedback phase, a NEU (13) is applied and additional QP iterations are conducted until pre-defined optimality and feasibility criteria hold. Subsequently, function values and derivatives are computed. The feedback phase is assumed to have negligible computational delay. Algorithmic improvements introduced by Würth et al. [17] in comparison to the work of Kadam and Marquardt [31] comprise the computation of the Hessian by composite adjoints [41], and the incorporation of a control grid adaptation strategy [60]. The authors applied the neighboring-extremal controller to the Williams-Otto continuous stirred-tank reactor (CSTR).

4.3.4. *Advanced-step controller by Zavala and Biegler [21]*

The authors introduce a discrete-time regulatory NMPC scheme with guaranteed stability. The inequality constraints are included in the objective function via barrier terms. In the online preparation phase, the previous solution is taken as a reference and OIS is applied. All function values and constraints are computed. In the feedback phase, a fast update is computed by SBU (7). The advanced step controller has been applied to an air separation unit [61], a low density polyethylene (LDPE) tubular reactor [62], an LDPE process [63], a distributed polymerization reactor [64] and a distillation column and a power plant [65].

4.3.5. *NMPC with incomplete optimization by Grüne and Pannek [66]*

The authors propose a fast discrete-time NMPC scheme with guaranteed stability. In the feedback phase, several QP iterations are conducted until derived stability criteria hold. It is assumed that the computational delay of the QP iterations is negligible. A similar scheme has been introduced by Graichen and Kugi [67] which has later been applied in the context of indirect solution approaches (see Section 6.2).

4.3.6. *Real-time nonlinear optimization as a generalized equation by Zavala and Anitescu [35]*

The authors introduce a fast sample-and-hold NMPC scheme guaranteeing stability with the help of generalized equations. While the online preparation phase does not exist, a suboptimal update (4) is approximated in the feedback phase with the help of an augmented Lagrangian reformulation. The resulting QP is solved with the help of projected successive over-relaxation. The iterative procedure is stopped after a pre-defined number is reached, usually before (4) is solved to convergence. As a case study, a nonlinear CSTR is considered.

4.3.7. *Best optimal control performance by Wolf et al. [34]*

The authors propose a fast sampled-data economic NMPC scheme without stability proof. It is assumed that the active set does not change. In the online preparation phase, the previous solution is taken as a reference and SIS is applied. All function values and derivatives are computed. In the feedback phase, a NEU (13) is applied and the controls are immediately sent to the process with negligible computational delay. Subsequently, the optimal number of QP iterations is conducted considering computational delay and using delay compensation by prediction. The delayed feedback information is once again sent to the process during the sampling interval. The neighboring-extremal controller has been applied to the Van de Vusse CSTR.

4.3.8. *Fast economic model predictive control by Jäschke et al. [28]*

A fast discrete-time economic NMPC scheme is introduced by Jäschke et al. [28] based on the work of Zavala and Biegler [21]. In the preparation phase, the previous solution is taken as a reference and OIS is used. In the feedback phase, Algorithm 2 is applied for the more general MFCQ to detect a change in the active set and avoid infeasibility. Hence, several parametric QPs (11) might be carried out which are assumed to have negligible computational delay. As a case study, a nonlinear CSTR is considered.

4.4. *Case study*

To compare the performance of different fast NMPC schemes, the operation of an isothermal CSTR as introduced by Diehl et al. [68] is considered with a single first-order, irreversible chemical reaction $A \rightarrow B$. The nonlinear system is given by

$$\begin{aligned}
\frac{dc_A}{dt} &= \frac{\dot{V}}{V}(c_{Af} - c_A) - kc_A,\\
\frac{dc_B}{dt} &= -\frac{\dot{V}}{V}c_B + kc_B,\\
\frac{dk}{dt} &= -0,
\end{aligned}$$

where c_A and c_B are the concentrations of raw material A and product B, respectively. k is the reaction constant. The flow rate $\dot{V}$ is the control variable. The parameters are set to $V = 10\,\text{l}$ and $c_{Af} = 1\,\text{mol/l}$ and the initial conditions are $c_{A,0} = 0.1\,\text{mol/l}$, $c_{B,0} = 0\,\text{mol/l}$ and $k_0 = 1.2\,\text{l/mol/min}$. The objective function is given by $\int_{t_{0,s}}^{t_{f,s}} (-2\dot{V}c_B\,1/\text{mol} + 0.5\dot{V}\,1/\text{l})\,d\tau$ considering the price of product B and a separation cost proportional to $\dot{V}$. The initial horizon length is 8 min. During operation, bounds are imposed on the concentrations and the flow rate with $0\,\text{mol/l} \le c_A \le 1\,\text{mol/l}$, $0\,\text{mol/l} \le c_B \le 1\,\text{mol/l}$, and $0\,\text{l/min} \le \dot{V} \le 20\,\text{l/min}$.

A sequential solution strategy is applied as described in [60] to compute the offline solution. The horizon is discretized into 16 equidistant control intervals. Thus, $n_\zeta = 16$. The path constraints are discretized on the same grid.

Though the operation of a CSTR calls for a moving horizon setting, a shrinking horizon setting is considered during online operation in order to exclude errors introduced by the prolongation of the horizon in a moving horizon setting. We assume that no unmeasurable disturbances exist. Furthermore, no plant-model mismatch and no measurement noise exist. At each sampling instant, all initial conditions are measured. The sampling period is $\Delta t = 1$ min. During online operation, k might change at each sampling instant in a step-wise fashion and stays constant otherwise. It increases from $k = 1.2\,\text{l/mol/min}$ to $k = 1.4\,\text{l/mol/min}$ at $t_{0,2} = 1$ min, to $k = 1.6\,\text{l/mol/min}$ at $t_{0,3} = 2$ min, to $k = 1.7\,\text{l/mol/min}$ at $t_{0,6} = 5$ min and to $k = 1.8\,\text{l/mol/min}$ at $t_{0,7} = 6$ min. Hereafter, k stays constant until the end of operation.

The fast NMPC schemes considered apply a NEU as a fast update method. The NEU is chosen because it improves suboptimal points and detects changes of the active set. The fast NMPC schemes differ in the online preparation phase: The first fast NMPC scheme (FS1) takes the nominal optimal solution as a reference and uses SIS. The second fast NMPC scheme (FS2) takes the solution of the previous horizon as a reference and applies SIS. The third fast NMPC scheme (FS3) takes the solution of the previous horizon as a reference and applies OIS. It is assumed that the computational delay of the feedback phase is negligible and that all computations of the online preparation phase including the computation of the optimal solution can be conducted within one sampling period. Hence, FS1, FS2 and FS3 are equally fast.

In Fig. 7, the predicted trajectories of the three fast NMPC schemes are shown for state variable c_A and control variable $\dot{V}$. It can be seen that all three schemes obtain similar predicted trajectories. Consequently, the values of the objective function are also similar. This is due to the fact that the fast update method applied provides a good update for this only slightly nonlinear optimal control problem independent of an optimal or suboptimal initial point. However, for optimal control problems with more severe nonlinearities, fast update methods do not generally provide a satisfactory update. In this case, the location of the initial point with respect to the optimal solution manifold becomes crucial which can be influenced by the choice of the reference and initialization strategy. In order to be able to compare the choice of the reference and the initialization strategy despite the good update for this example,

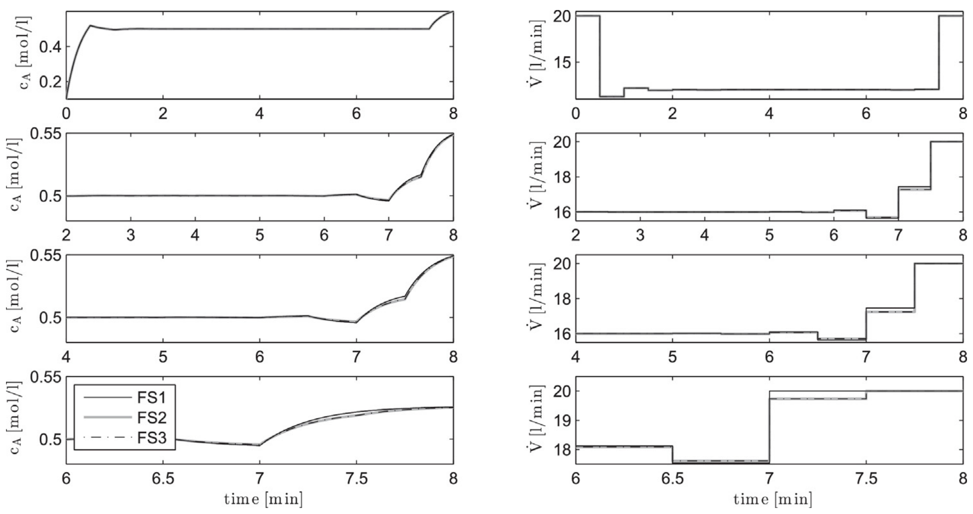

Fig. 7. Predicted trajectories of fast NMPC schemes. First row: predicted state and control variable at sampling instant $t_{0,1}$. Second row: predicted variables at sampling instant $t_{0,3}$. Third row: predicted variables at sampling instant $t_{0,5}$. Fourth row: predicted variables at sampling instant $t_{0,7}$.

Table 1
Optimality criterion ϵ_{opt} for predicted trajectories of fast NMPC schemes FS1, FS2 and FS3 at sampling instants.

	FS1	FS2	FS3
$t_{0,1}$	0	0	0
$t_{0,2}$	3.7×10^{-4}	3.7×10^{-4}	3.7×10^{-4}
$t_{0,3}$	1.3×10^{-3}	2.8×10^{-4}	2.7×10^{-4}
$t_{0,4}$	1.4×10^{-3}	3.3×10^{-7}	0
$t_{0,5}$	1.3×10^{-3}	0	0
$t_{0,6}$	1.9×10^{-3}	4.5×10^{-5}	4.5×10^{-5}
$t_{0,7}$	2.5×10^{-3}	7.8×10^{-5}	7.7×10^{-5}

Table 2
Optimality criterion ϵ_{opt} for predicted trajectories of fast NMPC schemes FS1-QP, FS2-QP and FS3-QP at sampling instants.

	FS1-QP	FS2-QP	FS3-QP
$t_{0,1}$	0	0	0
$t_{0,2}$	3.8×10^{-2}	3.8×10^{-2}	3.8×10^{-2}
$t_{0,3}$	7.3×10^{-2}	3.6×10^{-2}	3.2×10^{-2}
$t_{0,4}$	7.3×10^{-2}	3.4×10^{-3}	0
$t_{0,5}$	7.3×10^{-2}	5.0×10^{-5}	0
$t_{0,6}$	8.9×10^{-2}	1.4×10^{-2}	1.4×10^{-2}
$t_{0,7}$	1.0×10^{-1}	1.3×10^{-2}	1.3×10^{-2}

the following optimality criterion is introduced for the predicted trajectories at each sampling instant [17]:

$$\epsilon_{opt} = \frac{\|L_\zeta\|_\infty}{1+\|\lambda\|_2+\|\mu\|_2}. \tag{23}$$

As the optimality criterion evaluates the residual errors of the KKT conditions in the Lagrangian, smaller values are favorable. The difference in the performance of the fast NMPC schemes becomes apparent when looking at Table 1. It can be seen that FS1 performs worst: since p drifts away from p^{nom}, Δp becomes larger at each sampling instant and the update becomes less precise. FS2 tracks the optimal solution reasonably well. Since the previous solution is taken as a reference, FS2 is able to follow the disturbance trend though the initial point is slightly suboptimal at each sampling instant. As expected, FS3 has the best control performance because it starts from an initial point which was optimal at the previous sampling instant. Consequently, the error vanishes at sampling instants $t_{0,4}$ and $t_{0,5}$, because $\Delta p = 0$.

In order to illustrate the advantage of a fast update method that accounts for a change in the current state such as NEU, the NEU is replaced by a QP in the feedback phase of the three fast NMPC schemes. The respective schemes are denoted by FS1-QP, FS2-QP and FS3-QP. Table 2 shows the performance of these three fast NMPC schemes. It becomes apparent that FS1-QP performs worst because the nominal solution is already optimal. In contrast, FS2-QP and FS3-QP are able to follow the disturbance trend since the solution of the previous horizon is taken as a reference. FS3-QP shows again the best control performance. When comparing Tables 1 and 2, it can be seen, as expected, that the fast NMPC schemes applying a NEU perform better than the schemes relying on a QP. This is due to the fact that the QP cannot account for changes in the current state in contrast to the NEU.

4.5. Discussion

In the following, we will discuss which fast update method and which combination of elements of the fast NMPC promise best possible control performance.

To apply a fast update method other than QP (4) pays off if p can be reliably measured or estimated and if p is contained at least linearly in the constraints or bilinearly in the objective function as already mentioned in Section 3.7. In practice, this is already the case for linear MPC (LMPC) problems, where p solely includes initial conditions of the model states (cf. [27]). The most suitable fast update method also depends on the initial point, the number of active set changes expected and the computational time required. While suitable choices of fast update methods based on these three criteria were discussed in detail in Section 3.7, we will now transfer and evaluate these criteria in an online context. Whether computational delay is negligible depends on the ratio of solution time to sampling period. The larger the system considered and the faster the update rate of the measurements, the larger and thus more critical this ratio becomes. The number of active set changes to be expected depends strongly on the problem formulation, e.g. whether a regulatory objective or an economic objective has been chosen (cf. [8]). Finally, whether the initial point is close to the optimal solution is influenced by the initialization strategy chosen, the nonlinearity of the system and the disturbances acting on the process. Here, it is important to note that a fast update method that does not improve a suboptimal initial point such as SBU benefits from OIS because the controller is prevented from drifting away from the optimal solution manifold.

If no delayed feedback can be sent to the process during the sampling interval, it is best to use the previous solution as a reference and apply the OIS in the online preparation phase because the vicinity to the optimal solution manifold can be guaranteed independently of the nonlinearity of the system and the disturbances acting on the process. This is true whether or not the fast update method tracks the optimal solution under disturbances sufficiently well (cf. Section 4.4). A prerequisite is, however, that the optimal solution and the derivatives can be computed within one sampling period. If this is not possible, IIS can be applied to receive an improved initial point in comparison to SIS. If a new optimal solution is required, but it cannot be computed within one sampling interval, extended fast NMPC schemes have been developed, which will be introduced in Section 5.

If delayed feedback information can be sent to the process also within the sampling interval, it might be advantageous to perform additional calculations in the feedback phase after the fast update has been computed and the results have been sent to the process. If just one additional update is possible during the sampling interval, it is beneficial to conduct the optimal number of QP iterations for best control performance [34]. Subsequently, the optimal solution or an improved initial point can be computed for the next sampling instant by OIS and IIS, respectively. If an arbitrary number of updates can be sent to the process within the sampling interval, it might be advantageous to send updated information to the process after each QP iteration until the sampling instant $t_{0,s+1}$ is reached.

5. Extensions to fast NMPC schemes

Several extensions to the fast NMPC schemes presented in Section 4 exist. These extensions have been developed to frequently provide a new optimal solution in case the computation of the optimal solution is too involved to be finished within one sampling period. While these extensions still apply one of the fast update methods introduced in Section 3, they differ in their control architectures and in their fast NMPCs, respectively. Two classes of extensions to the fast NMPC scheme presented in Section 4 can be distinguished. While the first class makes use of a time-scale separation, the second class is based on parallel computing so that an optimal initial point can be provided at each sampling instant.

5.1. Time-scale separation

The fast NMPC schemes that belong to the time-scale separation class provide a fast update of the control variables at each sampling instant. The optimal solution to PNLP (2) is, however, provided (asynchronously) at a lower sampling rate. This time-scale separation can be either realized by employing a cascade-like hierarchical control architecture or by computing the optimal solution over several consecutive sampling intervals in the online preparation phase of a fast NMPC embedded in a single-layer control architecture.

In industrial process control, hierarchical architectures are routinely applied, where the upper-layer controller frequently provides a new reference to the lower-layer controller [4,6]. Based on the reference as well as current measurements, the lower-layer controller sends updated controls to the process at each sampling instant. Besides the controllers on each layer, a hierarchical architecture may comprise additional blocks such as an estimator as well as filters and triggers for the controllers. While a static real-time optimization is conventionally conducted on the upper layer (see, for example, [4]), Helbig et al. [2] and Backx et al. [69] suggested for the first time to apply instead an economic NMPC in order to operate the process in an economically optimal way at any time. Later, the conventional MPC on the lower layer was replaced by a fast NMPC by Kadam and Marquardt [31].

The control architecture of the two-layer fast NMPC scheme is depicted in Fig. 8. Here, the upper-layer NMPC frequently provides a new optimal solution to the lower-layer fast NMPC if required.

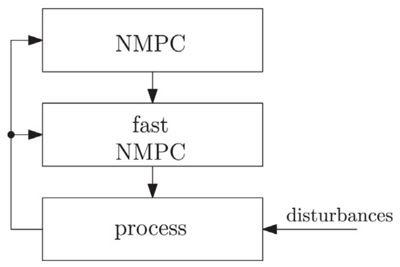

Fig. 8. Two-layer control architecture for fast NMPC schemes.

The lower-layer fast NMPC can realize all fast NMPC realizations presented in Section 4.

Besides hierarchical fast NMPC schemes, the time-scale separation class also comprises serial fast NMPC schemes based on a single-layer control architecture (see Fig. 4). To frequently provide an optimal solution, the online preparation phase of the fast serial NMPC spans several consecutive sampling intervals [29]. Thus, if the maximal computational delay to compute the optimal solution is $n\Delta t$, the optimal solution can be periodically provided at each nth sampling instant assuming negligible computational delay for the initialization strategy as well as for the fast update method. The reference corresponds to the last optimal solution provided. As an initialization strategy, DIS can be applied for a moving horizon setting and SIS for a shrinking horizon setting.

It is important to note that the hierarchical fast NMPC scheme can be considered as a generalization of the serial fast NMPC scheme. This is due to the fact that the hierarchical fast NMPC scheme reduces to the serial fast NMPC scheme if no additional blocks are employed, if the controllers of the two-layer control architecture are implemented on one core and if the last optimal solution provided by the NMPC is chosen to be the reference for the fast NMPC. If the computational delay of the initialization strategy and the fast update method are also negligible, the operational phase of the upper-layer NMPC corresponds to the online preparation phase of the fast serial NMPC, while the time intervals occupied by the operational phase of the lower-layer fast NMPC and the feedback phase of the fast serial NMPC tend to zero.

In the following, extensions to fast NMPC schemes are reviewed that are based on a time-scale separation.

5.1.1. A two-layer fast NMPC scheme by Kadam and Marquardt [31]

The first extended fast NMPC scheme has been presented by Kadam and Marquardt [31] (see also [70]). In this hierarchical fast sample-and-hold economic NMPC scheme, the upper-layer NMPC is triggered when optimality and feasibility criteria for the previous optimal solution are not met anymore. The NMPC provides an optimal solution to PNLP (2) as well as corresponding function values and derivatives. The lower-layer fast NMPC uses the upper-layer solution as a reference. In the online preparation phase, it applies SIS and recomputes the function values and first derivatives based on previous measurements, while the Hessian is not recomputed to reduce computational time. Then, NEU (13) is computed. In the feedback phase, the control variables computed in the previous online preparation phase are instantaneously applied to the process.

5.1.2. Real-time iterations on multiple layers by Bock et al. [13]

The authors propose a hierarchical fast sample-and-hold NMPC scheme without stability proof which is based on different variants of the real-time iterations. The control architecture consists of four layers: While the upper-layer controller performs standard real-time iterations, the three lower-layer controllers take the solution of their respective upper-layer controller as a reference, apply DIS and may recompute some of the information based on previous measurements before the RTI variants are performed: the controller on the third layer recomputes the function values as well as the objective gradient and the controller on the second layer recomputes the function values and approximates the objective gradient. A similar hierarchical fast NMPC scheme has been applied by Albersmeyer et al. [71], where solely an upper- and a lower-layer controller are used in an automotive application.

5.1.3. *A two-layer fast NMPC scheme by Würth et al. [72]*

The hierarchical fast sample-and-hold economic NMPC scheme is based on the work of Kadam and Marquardt [31]. The upper-layer controller is again triggered when optimality and feasibility criteria for the previous optimal solution no longer hold. In contrast to [31], a filter is applied to provide slow disturbances to the upper-layer NMPC and fast disturbances to the lower-layer fast NMPC. In order to account for computational delay that arises when computing the optimal solution to PNLP (2) as well as corresponding function values and derivatives, the upper-layer NMPC applies delay compensation by fixing the related decision variables in (2). To reduce computational delay, the Hessian is computed by composite adjoints [41]. The lower-layer fast NMPC uses the upper-layer solution as a reference and applies SIS in the online preparation phase. In the feedback phase, NEU provides a fast update.

5.1.4. *A two-layer fast NMPC scheme by Wolf et al. [8]*

The hierarchical fast sampled-data economic NMPC scheme extends the work of Würth et al. [72] to a special class of hybrid systems comprising integrated scheduling and control problems. As in [72], the disturbances are filtered. However, the upper-layer NMPC is only triggered if the optimality and feasibility criteria for the previous solution of the lower-layer fast NMPC do no longer hold. To account for computational delay, the upper-layer NMPC applies delay compensation by prediction, i.e., by simulating the model forward. The Hessian is again computed by composite adjoints [41]. The lower-layer fast NMPC either uses the upper-layer solution if available or its own previous solution as a reference and applies SIS in the online preparation phase. In the feedback phase, NEU provides a fast update.

5.1.5. *Hierarchical distributed neighboring-extremal controller by Wolf et al. [36]*

A hierarchical distributed fast sampled-data economic NMPC scheme is proposed by Wolf et al. [36] without stability proof. While the optimal solution is frequently recomputed on the upper layer, the lower-layer fast NMPC utilizes the optimal solution provided by the upper-layer NMPC as a reference and applies SIS. In the feedback phase, a distributed version of the NEU (13) is used. A similar fast NMPC scheme has been suggested later by Ruben et al. [73].

5.1.6. *Serial approach of the advanced-multi-step controller by Yang and Biegler [29]*

The authors propose a serial fast discrete-time regulatory NMPC scheme including a nominal stability proof. In the online preparation phase, the optimal solution is taken as a reference and SIS is applied. In the feedback phase, the update is computed by SBU (7) and changes of the active set are handled by an additional QP iteration.

5.2. *Parallel computing*

In order to provide an optimal solution at each sampling instant, Yang and Biegler [29] suggested to make use of several processors or cores of a processor in parallel such that several fast NMPCs work in parallel in an alternated way. Each fast NMPC makes use of delay compensation to determine the predicted state over more than one sampling period.

Table 3
Optimality criterion ϵ_{opt} for predicted trajectories of fast NMPC schemes at sampling instants for first scenario with fluctuating reaction constant.

	FS1	FS2
$t_{0,1}$	0	0
$t_{0,2}$	3.7×10^{-4}	3.7×10^{-4}
$t_{0,3}$	0	3.4×10^{-4}
$t_{0,4}$	4.3×10^{-4}	4.6×10^{-4}
$t_{0,5}$	0	5.0×10^{-4}
$t_{0,6}$	3.6×10^{-4}	3.7×10^{-4}
$t_{0,7}$	4.9×10^{-4}	3.3×10^{-4}

5.2.1. *Parallel approach of the advanced-multi-step controller by Yang and Biegler [29]*

The authors propose a parallel fast discrete-time regulatory NMPC scheme including a nominal stability proof. If the maximal computational delay is $n\Delta t$, n processors are applied. For each processor, the following steps are carried out: At sampling instant $t_{0,s}$, compute the optimal solution based on the predicted state $\bar{x}(t_{0,s} + n\Delta t)$. At sampling instant $t_{0,s+n} = t_{0,s} + n\Delta t$, get the current measurements. Compute SBU (7) based on current measurements as well as the optimal solution computed during the time interval $[t_{0,s}, t_{0,s} + n\Delta t]$. Immediately send the updated controls to the process. If $n \leq 1$, the parallel approach reduces to the advanced-step controller suggested by Zavala and Biegler [21], which has been reviewed in Section 4.3.

5.3. *Case study*

In this subsection, we compare the performance of an extended fast NMPC scheme with one of the fast NMPC schemes introduced in Section 4. The optimal operation of the isothermal CSTR as described in Section 4.4 will again be considered. For both fast NMPC schemes, NEU (13) is chosen as a fast update method assuming negligible computational delay. The first fast NMPC scheme (FS1) is a serial fast NMPC scheme. As hierarchical fast NMPC schemes can achieve the same control performance as the serial fast NMPC scheme (cf. Section 5.1), the serial fast NMPC scheme is chosen as a representative of the time-scale separation class. The optimal solution is provided at each third sampling instant and SIS is used. It is assumed that the computational delay of the initialization strategy is negligible. The second fast NMPC scheme (FS2) is also based on a single-layer control architecture, where the fast NMPC uses its previous solution as a reference and applies SIS in the online preparation phase. In the following, the performance of these three fast NMPC scheme are analyzed for three disturbance scenarios.

5.3.1. *First scenario: fluctuating reaction constant*

In the first scenario, k changes at each sampling instant in a step-wise fashion and stays constant otherwise. It changes to 1.2 l/mol/min at all uneven sampling instants and, alternately, to 1.4 l/mol/min and 1.0 l/mol/min at even ones. The performance of the fast NMPC schemes is shown in Table 3, where the optimality criterion (23) for the predicted trajectories is depicted. It can be seen that FS1 tracks the optimal solution under fluctuating disturbances slightly better than FS2. This is due to the fact that FS1 does not follow the fluctuating disturbances at each sampling instant immediately.

5.3.2. *Second scenario: jump in reaction constant*

In the second scenario, k stays constant at 1.2 l/mol/min from $[t_{0,1}, t_{0,4})$ and then jumps to 1.9 l/mol/min at $t_{0,4}$. The performance

Table 4
Optimality criterion ϵ_{opt} for predicted trajectories of fast NMPC schemes at sampling instants for second scenario with jump in reaction constant.

	FS1	FS2
$t_{0,1}$	0	0
$t_{0,2}$	0	0
$t_{0,3}$	0	0
$t_{0,4}$	3.9×10^{-3}	3.9×10^{-3}
$t_{0,5}$	3.9×10^{-3}	7.9×10^{-6}
$t_{0,6}$	3.9×10^{-3}	1.1×10^{-7}
$t_{0,7}$	0	0

of the fast NMPC schemes is shown in Table 4. It becomes apparent that FS2 tracks the optimal solution better than FS1. This is due to the fact that FS2 applies its previous solution as a reference and thus incrementally reduces the gap between its fast update and the optimal solution. In contrast, the fast update method of FS1 uses the optimal solution based on outdated state information for three consecutive sampling instants. Since delay compensation by prediction is applied based on an assumed constant reaction rate, the optimal solution at $t_{0,4}$ is based on $k = 1.2$ l/mol/min and the optimal solution at $t_{0,7}$ is based on $k = 1.9$ l/mol/min.

5.3.3. *Third scenario: trend in reaction constant*

In the third scenario, k increases by 0.1 l/mol/min at each sampling instant in a step-wise fashion starting from 1.2 l/mol/min. The results shown in Table 5 reveal that FS2 again tracks the optimal solution slightly better than FS1. This is due to the fact that FS2 applies its previous and almost optimal solution as a reference and thus follows the disturbance trend immediately. The fast update method of FS1, however, uses the optimal solution based on outdated state information. The optimal solution at $t_{0,4}$ is based on $k = 1.2$ l/mol/min and the optimal solution at $t_{0,7}$ is based on $k = 1.5$ l/mol/min. Consequently, FS1 cannot follow the disturbance trend instantaneously.

5.4. *Discussion*

The serial fast NMPC scheme has evolved from the standard fast NMPC scheme presented in Section 4. Since it provides an optimal solution every *nth* sampling instant, fast update methods can be applied which do not improve a suboptimal initial point (such as SBU). If a fast update method improving a suboptimal initial point is used as in the case study of Section 5.3, the serial fast NMPC scheme might not improve control performance compared to a standard fast NMPC scheme. This is due to the fact that it is more favorable to make use of a fast suboptimal feedback based on previous state information than an optimal solution computed for an outdated state. Consequently, the suitability of the single-layer fast NMPC schemes chosen depends strongly on the disturbances acting on the process. For fluctuating disturbances, it can be favorable to use the (outdated) optimal solution as a reference and to apply SIS. For disturbance jumps, it might be necessary to recompute the optimal solution because otherwise the fast NMPC scheme will not converge to the optimal solution manifold again. If a disturbance jump is known a priori based on disturbance predictions, the new optimal solution should be precomputed so that it can be provided when the disturbance jump affects the system. For disturbance trends, it is advantageous to use the previous solution as a reference as shown in Section 5.3. Here, IIS can further improve control performance in comparison to SIS as discussed in Section 4.5. For the considerations above, it was assumed that no delayed feedback can be sent to the process during the sampling interval. However, if delayed feedback can be sent to the process and if it is advantageous to use a standard fast NMPC scheme, the recommendations of Section 4.5 hold.

Though the partition into different hierarchical layers may reduce control performance in case of negligible computational delay [74], a hierarchical fast NMPC scheme promises an improved control performance in comparison to a fast NMPC scheme based on a single-layer architecture if computational delay has to be taken into account. If the computational delay is smaller than the sampling period, if no additional blocks such as filters are realized and if the controllers of the two-layer control architecture are implemented on one core, the hierarchical fast NMPC scheme reduces to a standard fast NMPC scheme. In case the computational delay exceeds one sampling period, the hierarchical fast NMPC scheme can achieve a control performance that is as good as the one of the serial fast NMPC scheme because the hierarchical fast NMPC scheme is a generalization of the serial one (see Section 5.1). However, further improvements can be expected if each controller of the hierarchical fast NMPC scheme is implemented on its own core. This is due to the fact that the lower-layer fast NMPC can be chosen such that any combination of elements presented in Section 4 can be realized, while a new optimal solution can be provided latest after n sampling instants by the upper-layer NMPC even though the computational delay of the feedback phase of the fast NMPC might not be negligible. In this way, disturbance trends can be effectively followed as also shown by Wolf et al. [8].

The parallel fast NMPC scheme of Yang and Biegler [29] improves the serial fast NMPC scheme of the same authors in case an optimal solution is required more frequently for stability or performance reasons. It promises good control performance for fluctuating disturbances. After a disturbance jump, the new optimal solution is provided after n sampling instants as it is also the case for a serial fast NMPC scheme as well as for a hierarchical fast NMPC scheme. However, the optimal solution provided by the parallel fast NMPC scheme at the next sampling instant might by chance be sufficient to effectively treat the disturbance jump directly. A drawback of the parallel approach is that disturbance trends are only followed with a delay of n sampling instants in contrast to a hierarchical fast NMPC scheme, where the lower-layer fast NMPC can follow disturbance trends in a suboptimal way at each sampling instant.

6. Closely related NMPC schemes

Many additional NMPC schemes exist which are closely related to the fast NMPC schemes presented in Section 4. Within this group of closely related NMPC schemes, several subgroups which will be presented in chronological order can be identified. First, NMPC schemes are presented which apply updates based on a linearization of the model equations. Second, NMPC schemes are outlined which rely on fast update methods developed for indirect solution approaches. Third, explicit MPC schemes are considered, where the main computational effort is shifted to the offline preparation. Finally, continuation-based NMPC schemes are presented.

Table 5
Optimality criterion ϵ_{opt} for predicted trajectories of fast NMPC schemes at sampling instants for third scenario with trend in reaction constant.

	FS1	FS2
$t_{0,1}$	0	0
$t_{0,2}$	1.0×10^{-4}	1.0×10^{-4}
$t_{0,3}$	3.7×10^{-4}	8.3×10^{-5}
$t_{0,4}$	8.3×10^{-4}	7.4×10^{-5}
$t_{0,5}$	1.5×10^{-3}	6.4×10^{-5}
$t_{0,6}$	1.9×10^{-3}	4.7×10^{-5}
$t_{0,7}$	4.5×10^{-4}	7.8×10^{-5}

6.1. NMPC schemes based on linearization of model

NMPC schemes which are based on a linearization of the model have already been applied since the early nineties. These schemes have provided an important basis for the development of the fast NMPC schemes described in Section 4 because only one or few QP iterations are conducted in the feedback phase as it is done for the fast NMPC schemes relying on a suboptimal update method. However, the method of deriving the QP is different, which will be outlined next.

All NMPC schemes based on a model linearization are developed for a special type of optimal control problems, where the objective function is quadratic or linear, the path constraints are box constraints and no endpoint constraints are present. Before or after the discretization of the controls and the equations of (1), the nonlinear model is linearized either around the current operating point or a given reference trajectory. The discretized and linearized equations of (1) directly result in a QP or an LP. This is in contrast to the fast NMPC schemes considered in Section 4, where PNLP (2) is obtained and the related KKT system (3) is linearized. Since the model linearization is carried out after measurements have been taken, several computations must be conducted in the feedback phase such that computational delay is often not negligible.

6.1.1. Newton-type control strategy by Li and Biegler [75]

The authors propose a sample-and-hold regulatory NMPC scheme without stability considerations, which tracks a pre-specified setpoint. All computations are carried out in the feedback phase. A first-order correction of the outputs is computed iteratively starting from the reference solution until the change in the objective function falls below a pre-defined threshold. Alternatively, the authors suggest to conduct a pre-defined number of iterations. The reference is taken to be the control trajectory computed in the previous iteration. For the first iteration, the control applied previously to the process is taken as a reference for the entire time horizon. Oliveira and Biegler [76] extend the scheme such that reference trajectories can be tracked and a shift initialization is conducted.

6.1.2. Nonlinear quadratic dynamic matrix control by Gattu and Zafiriou [77]

The authors propose a sample-and-hold regulatory NMPC scheme without stability considerations. In the online preparation phase, a state estimation is conducted. In the feedback phase, the model is linearized around the current reference trajectory and discretized. The predicted outputs are computed taking two influences into account: the effect of future manipulated variables is estimated by step response coefficients and the effect of past measurements is approximated by a forward simulation. Subsequently, a QP is conducted.

6.1.3. Constrained receding horizon predictive control by Lee et al. [78]

A discrete-time regulatory NMPC scheme is introduced by the authors including a stability proof. In the online preparation phase, a new reference trajectory is generated via SIS of the old reference trajectory. In the feedback phase, the model is linearized around the reference trajectory and a linear objective is defined with the help of the bounded error between the predicted state and the reference solution. Subsequently, a linear program is solved.

6.1.4. Model predictive control algorithm by Tiagounov and Weiland [79]

The authors propose a sample-and-hold regulatory NMPC scheme without stability proof. All computations are carried out in the feedback phase. Here, the reference trajectory is computed by integrating the nonlinear model forward given the control variables from the previous horizon. Then, the model is linearized and discretized around the reference trajectory. Subsequently, a QP is conducted.

6.2. NMPC schemes based on fast update method for indirect solution approach

NMPC schemes based on fast update methods developed for indirect solution approaches have already been published in the late eighties and have strongly influenced the development of similar schemes based on direct solution approaches. In the following, indirect solution approaches as well as their fast update methods are outlined briefly.

The optimal solution to (1) satisfies Pontryagin's maximum principle, which is a necessary condition for an optimal solution [80]. Pontryagin's maximum principle is given by a multi-point boundary value problem (MPBVP) with interior switching points, which denote the time points when the active set of constraints changes. To solve the MPBVP to convergence, iterative indirect solution methods exist, for which the sequence of arcs, i.e., the time intervals with a constant active set, must be guessed a priori. We refer the interested reader to the review by Binder et al. [81] for more details on direct solution approaches. In order to receive a fast update, suboptimal and sensitivity-based update methods exist for indirect solution approaches comparable to the ones of the direct solution approaches (cf. Section 3).

6.2.1. Neighboring extremals by Pesch [82]

Already in the late eighties, the author introduced a sample-and-hold NMPC scheme without a stability proof. It is assumed that the sequence of arcs is not changed throughout the operation. In the offline preparation, the optimal reference trajectory is computed by solving the MPBVP. While the online preparation phase does not exist, the feedback phase spans over an entire sampling interval because pre-calculation of switching points and feasibility checks are carried out after the sensitivity-based update, the so-called neighboring extremal, has been received. To compensate the arising computational delay, delay compensation by prediction is used to compute the state at the next sampling instant. According to the author, a simplified version without checks and pre-calculations has been applied, which has negligible computational delay.

6.2.2. Suboptimal MPC by Graichen and Kugi [67]

A sample-and-hold NMPC scheme is introduced by the authors including a stability proof. In the online preparation phase, the previous solution is taken as a reference and SIS is applied. In the feedback phase, a user-defined optimization algorithm with linear convergence is used. Several iterations are conducted with assumed negligible computational delay until a lower iteration number for exponential stability is reached. Graichen et al. [83] suggest to solve the MPBVP with the help of a projected gradient method for control-constrained systems. Later, Graichen [84] develops a fix-point scheme for input-affine systems with control constraints resulting in two integrations per iteration. Suboptimal MPC has been applied to a crane [67], to an inverted pendulum [85], to a Van de Vusse CSTR and to a magnetic levitation train [86].

6.3. Explicit MPC

Explicit MPC is a fast NMPC scheme where the main computational effort is shifted to offline preparation. During offline preparation, a table of so-called critical regions and corresponding affine control functions is computed by multi-parametric nonlinear programming (mp-NLP). Mp-NLP approximates the optimal solution of (2) under parametric changes for $p \in \mathbb{R}^{n_p}$ and can thus be considered as a generalization of NLPP (see Section 3.6), where only a scalar parameter is considered. Early results on mp-NLP go back to [20] (cf. Section 3.2) and [87]. In mp-NLP, the decision variables are approximated as affine functions of the parameters in subspaces of the parameter space P. These subspaces are termed critical regions which are determined either such that error bounds related to the objective function and the constraints hold or by successive linearization of the nonlinear constraints. An overview of the algorithms applied for mp-NLP is given by Domínguez et al. [88]. An important special case is multi-parametric quadratic programming (mp-QP), because no approximation error exists. If it is assumed that the Hessian is positive definite, if the linear independence constraint qualifications hold and if $p \in P$, where P is convex, the set of feasible parameters P_f is convex and the optimizer $\zeta^* : P_f \to \mathbb{R}^{n_\zeta}$ is continuous and piecewise affine (cf. Theorem 4 in [89]). An algorithm determines the finite number of so-called critical regions, where the active set is constant and ζ^* is affine, together with the affine function ζ^* [90]. After the optimal solution ζ^{init^*} has been determined, the critical region for a given p^{init} is determined with the help of functions similar to (9) and (10). A drawback of mp-QP, and mp-NLP in particular, is that the methods are only suited for small-scale problems because the number of critical regions to be explored grows significantly with the number of inequality constraints. For large-scale problems, the offline computation is typically too involved and the table look-up of the critical regions during the feedback phase will take too long for a fast update. To overcome these problems, Pannocchia et al. [91] suggested to combine explicit MPC with online optimization methods (see also the related works [92–94]). In the following, we will describe important fast NMPC schemes for explicit LMPC, explicit NMPC and explicit MPC combined with online optimization methods. For a detailed overview on explicit LMPC, we refer the interested reader to the work of Alessio and Bemporad [95] and for an overview on explicit NMPC to the work of Domínguez and Pistikopoulos [96].

6.3.1. *Explicit LMPC by Pistikopoulos et al. [90]*

The authors present a fast discrete-time LMPC scheme without stability proof. In the offline preparation, the optimal control problem is reformulated to an mp-QP and an algorithm is applied such that the critical regions and corresponding affine optimal functions ζ^* are determined and stored. While the online preparation phase does not exist, the critical region is determined in the feedback phase and the corresponding affine control function is evaluated at p_s. Bemporad et al. [89] extend this fast NMPC scheme by a stability proof, an improved determination of the critical regions and an online linear search through the critical regions. Furthermore, Bemporad et al. [97] consider the case where the mp-QP reduces to a multi-parametric linear problem.

6.3.2. *Explicit NMPC by Johansen [98]*

The author suggests a fast discrete-time NMPC scheme without stability proof. In the offline preparation, a local mp-QP is solved and the errors in the controls, in the objective and in the constraints are estimated. If the error is too high, the region is subdivided and further local mp-QPs are solved. Finally, the critical regions and the corresponding affine functions ζ are stored. The online preparation phase and the feedback phase correspond to the one of the explicit LMPC scheme. Later, Johansen [99] also proved asymptotic stability.

6.3.3. *Partial enumeration by Pannocchia et al. [91]*

The authors introduce a fast discrete-time LMPC scheme including a stability proof. In the offline preparation, an algorithm is applied for the mp-QP, where the critical regions and corresponding affine optimal functions ζ^* are determined and stored, but only for a small subset of all possible active sets. The online preparation phase depends on the last feedback phase. If the active set was enclosed in the table, the information when and how often this active set is used is updated to speed up the table search. Otherwise, a new optimal solution is computed until the next sampling instant and the table is updated. In the feedback phase, the table is searched to find the current active set. Subsequently, the corresponding affine control function is evaluated at p_s. If the current active set was not found, the shifted solution from the last horizon is applied. However, if the shifted solution does not ensure satisfactory control performance, the optimal solution is determined for a short horizon. To prove robust exponential stability, Pannocchia et al. [100] suggested an improved fast update method in case the active set is not contained in the table.

6.4. *NMPC schemes based on continuation*

Section 3.6 illustrated the close relation between NLPP, which is a subfield of continuation theory, and the sensitivity-based update methods used in fast NMPC schemes. However, not only the concepts of NLPP have been applied but also general continuation theory is used in NMPC schemes to provide a fast update of the control variables at each sampling instant with negligible computational delay.

6.4.1. *Continuation/generalized minimum residual (GRMES) update method by Ohtsuka [101]*

The author suggests a sample-and-hold NMPC scheme with a stability proof based on an error analysis. The inequality constraints are included in the objective function via barrier terms and an indirect solution approach is applied making use of a single-shooting strategy. While the online preparation phase does not exist, a fast continuation-based update is computed in the feedback phase which tracks a system of nonlinear equations (20). The system contains the discretized equality constraints as well as the discretized first-order necessary conditions for the control variables, $H_u = 0$. y comprises the discretized control variables and the corresponding multipliers. p represents the process time t and the current state of the system which depends on t. The update is computed by a predictor step similar to (21). The NMPC scheme has been extended to multiple shooting by Shimizu et al. [102], to descriptor systems by Marutani and Ohtsuka [103] and to infinite-horizon optimal control by Marutani and Ohtsuka [104]. The continuation/GRMES update method has been applied to control hovercrafts [105,102], a four-wheel vehicle and an inverse pendulum [103].

6.4.2. *Real-time framework by DeHaan and Guay [106]*

The authors introduce a sampled-data NMPC scheme with stability proof which is closely related to continuation and adaptive control. The inequalities are included in the objective via barrier terms. The control variables are parameterized on a grid and the grid points are denoted as the switching times. While the online preparation phase does not exist, a differential equation for the control parameters and the switching times is continuously integrated and updated based on the current p_s in the feedback phase. If a switching time is reached, the differential equation is reinitialized and the integration is restarted. Extensions to the real-time framework have been suggested by DeHaan and Guay [107].

7. Conclusions

In this review, we compared fast NMPC schemes that have been developed to provide a fast but typically suboptimal update for the solution to the optimal control problem at each sampling instant. By providing fast updates, the computational delay of the feedback phase is significantly reduced such that it often even becomes negligible. Hence, these fast NMPC schemes allow to eliminate one of the main hindrances for large-scale NMPC applications in the process industry today.

We first focused on two subclasses of fast update methods developed for direct solution approaches, the suboptimal update methods and the sensitivity-based update methods. It was shown that the choice of the fast update method to be applied in the feedback phase depends on the problem formulation, the nonlinearity and the size of the system, the update rate of the measurements, the initialization strategy chosen and the disturbances acting on the system. Here, it should be mentioned that, in case disturbances cannot be measured or estimated, they cannot be directly accounted for by fast update methods. Besides the fast update method applied, the elements of the fast NMPC are crucial for control performance. The best suited fast NMPC scheme depends on the ratio of the computational time required to solve the optimal control problem to convergence and the sampling period. Furthermore, it depends on how often a control update can be sent to the process within one sampling interval. Finally, safety and reliability considerations also affect the choice of the fast NMPC scheme.

As depicted by the illustrative example and the case studies, it is often advantageous to apply a fast update method which can inherently detect changes of the active set, improve a suboptimal initial point and predict the change of the optimal solution with respect to the current initial condition in a path-following manner. Though the computational delay arising for these particular methods is higher than for other fast update methods, efficient approximations and algorithmic improvements allow for a feedback within the range of microseconds for smaller scale processes. If the computational time to solve the optimal control problem to convergence is smaller than one sampling period, the most suitable fast NMPC scheme is the one that is based on a single-layer architecture, that uses the previous solution as a reference and applies OIS. In case the computational time exceeds one sampling period, a serial fast NMPC scheme or a parallel approach might outperform the standard fast NMPC scheme depending on the disturbances acting on the process. However, both the serial fast NMPC scheme and the parallel approach have the disadvantage that disturbance trends are only followed with a delay of n sampling instants. This shortcoming can be avoided by applying the following two-layer fast NMPC scheme, which we recommend to ensure best possible control performance: while the lower-layer fast NMPC uses its own previous solution as a reference and applies IIS to follow disturbance trends immediately and to compensate fluctuating disturbances, the upper-layer NMPC treats severe disturbances such as jumps. If the severe disturbances are known a priori, the optimal solution can be computed beforehand and can be sent to the lower-layer fast NMPC when the disturbances occur. Otherwise, the disturbances are detected applying a filter and a trigger as in the work of Wolf et al. [8] and the optimal solution will be sent to the lower-layer controller as soon as available. Hereafter, the lower-layer fast NMPC replaces its own previous solution by the optimal solution of the upper-layer NMPC. Besides the benefit of also treating disturbance trends directly, the suggested two-layer fast NMPC scheme has the vital benefit of improving the safety and the reliability of the process.

In this unifying review, it was shown that similar fast NMPC schemes have been developed over the last 15 years in literature. The similarities between these schemes are based on the fact that all schemes rely on few common elements regarding their fast NMPC and that they employ similar fast update methods as well as control architectures. A lot of fast NMPC schemes have been suggested because the number of possible combinations of these elements is high and the number of fast NMPC schemes even increases if these elements are combined with one of the fast update methods. Since there is no additional use in simply combining elements of the fast NMPC and fast update methods to a new fast NMPC scheme, ideally, the benefits of a new fast NMPC scheme should be proven compared to schemes that have already been published. To achieve this, detailed studies are required that analyze and compare the performance of fast NMPC schemes. Based on these studies, the most suitable fast NMPC scheme should be chosen for a specific application and scenario at hand.

References

[1] I.J. Wolf, Economic model-predictive control for chemical processes (Ph.D. thesis), RWTH Aachen University, Shaker Verlag, 2016.
[2] A. Helbig, O. Abel, W. Marquardt, Structural concepts for optimization based control of transient processes, in: F. Allgöwer, A. Zheng (Eds.), Nonlinear Model Predictive Control Workshop – Assessment and Future Directions, Ascona, Switzerland, 3–5.6.1998, vol. 26, Birkhäuser Verlag, Basel, 2000, pp. 295–311.
[3] J.B. Rawlings, R. Amrit, Optimizing process economic performance using model predictive control, in: L. Magni, D. Raimondo, F. Allgöwer (Eds.), Nonlinear Model Predictive Control: Towards New Challenging Applications, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 119–138.
[4] S.J. Qin, T.A. Badgwell, A survey of industrial model predictive control technology, Control Eng. Pract. 11 (2003) 733–764.
[5] R. Findeisen, F. Allgöwer, Computational delay in nonlinear model predictive control, in: Proc. Int. Symp. Adv. Control of Chemical Processes, IFAC ADCHEM, vol. 3, 2004, pp. 427–432, URL Paper ID LTD 4.1.5 on CD-ROM.
[6] R. Scattolini, Architectures for distributed and hierarchical model predictive control – A review, J. Process Control 19 (5) (2009) 723–731.
[7] M. Ellis, P.D. Christofides, Integrating dynamic economic optimization and model predictive control for optimal operation of nonlinear process systems, Control Eng. Pract. 22 (2014) 242–251.
[8] I.J. Wolf, D.A. Muñoz, W. Marquardt, Consistent hierarchical economic NMPC for a class of hybrid systems using neighboring-extremal updates, J. Process Control 24 (2014) 389–398.
[9] J. Zhang, S. Liu, J. Liu, Economic model predictive control with triggered evaluations: state and output feedback, J. Process Control 24 (2014) 1197–1206.
[10] M. Ellis, P.D. Christofides, Real-ime economic model predictive control of nonlinear process systems, AIChE J. 61 (2015) 555–571.
[11] P.O.M. Scokaert, D.Q. Mayne, J.B. Rawlings, Suboptimal model predictive control (feasibility implies stability), IEE Trans. Autom. Control 44 (1999) 648–654.
[12] M. Ellis, H. Durand, P.D. Christofides, A tutorial review of economic model predictive control methods, J. Process Control 24 (2014) 1156–1178.
[13] H.G. Bock, M. Diehl, E. Kostina, J.P. Schlöder, Constrained optimal feedback control of systems governed by large differential algebraic equations, in: L. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes, S.B. van Bloemen Waanders (Eds.), Real-time PDE-Constrained Optimization, SIAM, Philadelphia, 2007, pp. 3–24.
[14] M. Diehl, H. Ferreau, N. Haverbeke, Efficient numerical methods for nonlinear MPC and moving horizon estimation, in: L. Magni, D. Raimondo, F. Allgöwer (Eds.), Nonlinear Model Predictive Control, Springer, 2009, pp. 391–417.
[15] L.T. Biegler, Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes, Siam, 2010.
[16] J. Nocedal, S. Wright, Numerical Optimization, Springer, 1999.
[17] L. Würth, R. Hannemann, W. Marquardt, Neighboring-extremal updates for nonlinear model-predictive control and dynamic real-time optimization, J. Process Control 19 (8) (2009) 1277–1288.
[18] M. Diehl, H. Bock, J. Schlöder, R. Findeisen, Z. Nagy, F. Allgöwer, Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations, J. Process Control 12 (4) (2002) 577–585.
[19] A.V. Fiacco, G.P. McCormick, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley & Sons, Inc., 1968.
[20] A.V. Fiacco, Sensitivity analysis for nonlinear programming using penalty methods, Math. Program. 10 (1976) 287–311.
[21] V. Zavala, L. Biegler, The advanced-step NMPC controller: optimality, stability and robustness, Automatica 45 (2009) 86–93.
[22] J.H. Bigelow, N.Z. Shapiro, Implicit function theorems for mathematical programming and for systems of inequalities, Math. Program. 6 (1974) 141–156.
[23] A. Shapiro, Sensitivity analysis of nonlinear programs and differentiability properties of metric projections, SIAM J. Control Optim. 26 (1988) 628–645.
[24] T.J. Beltracchi, G.A. Gabriele, Observations on extrapolations using parameter sensitivitiy derivatives, Adv. Des. Autom. 14 (1988) 165–174.
[25] C. Büskens, H. Maurer, Sensitivity analysis and real-time optimization of parametric nonlinear programming problems, in: M. Grötschel, S.O. Krumke, J. Rambau (Eds.), Online Optimization of Large Scale Systems: State of the Art, Springer-Verlag, 2001, pp. 3–16.
[26] M. Kojima, R. Hirabayashi, Continuous deformation of nonlinear programs, Sens. Stabil. Parametr. Anal. 21 (1984) 150–198.

[27] H. Ferreau, H. Bock, M. Diehl, An online active set strategy to overcome the limitations of explicit MPC, Int. J. Robust Nonlinear Control 18 (8) (2008) 816–830.
[28] J. Jäschke, X. Yang, L.T. Biegler, Fast economic model predictive control based on NLP-sensitivities, J. Process Control 24 (2014) 1260–1272.
[29] X. Yang, L.T. Biegler, Advanced-multi-step nonlinear model predictive control, J. Process Control 23 (2013) 1116–1128.
[30] V.M. Zavala, Computational strategies for the optimal operation of large-scale chemical processes (Ph.D. thesis), Carnegie Mellon University, ProQuest, 2008.
[31] J. Kadam, W. Marquardt, Sensitivity-based solution updates in closed-loop dynamic optimization, in: Proc. of the IFAC DYCOPS, vol. 7, Cambridge, USA, 2004.
[32] J. Guddat, F. Guerra Vasquez, H.T. Jongen, Parametric Optimization: Singularities. Pathfollowing and Jumps, B.G. Teubner, Stuttgart, 1990.
[33] N. Ganesh, L.T. Biegler, A reduced Hessian strategy for sensitivity analysis of optimal flowsheets, AIChE J. 33 (2) (1987) 282–296.
[34] I.J. Wolf, L. Würth, W. Marquardt, Rigorous solution vs. fast update: acceptable computational delay in NMPC, in: Proc. of the 50th IEEE CDC-ECC, 2011, pp. 5230–5235.
[35] V.M. Zavala, M. Anitescu, Real-time nonlinear optimization as a generalized equation, SIAM J. Control Optim. 48 (2010) 5444–5467.
[36] I.J. Wolf, H. Scheu, W. Marquardt, A hierarchical distributed economic NMPC architecture based on neighboring-extremal updates, in: Proc. of ACC, Montréal, Canada, 2012, pp. 4155–4160.
[37] P. Seferlis, A. Hrymak, Sensitivity analysis for chemical process optimization, Comput. Chem. Eng. 20 (10) (1996) 1177–1200.
[38] E. Allgower, K. Georg, Introduction to Numerical Continuation Methods, vol. 45, Society for Industrial and Applied Mathematics, 2003.
[39] B. Houska, H.J. Ferreau, M. Diehl, An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range, Automatica 47 (2011) 2279–2285.
[40] R. Findeisen, T. Raff, F. Allgöwer, Sampled-data nonlinear model predictive control for constrained continuous time systems, in: Advanced Strategies in Control Systems with Input and Output Constraints, Lecture Notes in Control and Information Sciences, vol. 346, Springer, 2007, pp. 207–235.
[41] R. Hannemann, W. Marquardt, U. Naumann, B. Gendler, Discrete first- and second-order adjoints and automatic differentiation for the sensitivity-analysis of dynamic models, Proc. Comput. Sci. 1 (2012) 297–305.
[42] H.G. Bock, M. Diehl, D.B. Leineweber, J.P. Schlöder, A direct multiple shooting method for real-time optimization of nonlinear DAE processes, in: F. Allgöwer, A. Zheng (Eds.), Nonlinear Model Predictive Control, Progress in Systems and Control Theory, vol. 26, Birkhäuser Verlag, 2000, pp. 245–267.
[43] M. Alamir, Nonlinear receding horizon sub-optimal guidance law for the minimum interception time problem, Control Eng. Pract. 9 (2001) 107–116.
[44] R. Cagienard, P. Grieder, E. Kerrigan, M. Morari, Move blocking strategies in receding horizon control, J. Process Control 17 (2007) 563–570.
[45] R. Bellman, Dynamic Programming, Princeton UP, NJ, 1957.
[46] L. Grüne, Economic receding horizon control without terminal constraints, Automatica 49 (2013) 725–734.
[47] L. Würth, W. Marquardt, Infinite-horizon continuous-time NMPC via time transformation, IEEE Trans. Autom. Control 59 (2014) 2543–2548.
[48] A. Schäfer, P. Kühl, M. Diehl, J. Schlöder, H.G. Bock, Fast reduced multiple shooting methods for nonlinear model predictive control, Chem. Eng. Process. 46 (2007) 1200–1214.
[49] L. Wirsching, J. Albersmeyer, P. Kühl, M. Diehl, H.G. Bock, An adjoint-based numerical method for fast nonlinear model predictive control, in: Proc. of the 17th World Congress, Seoul, Korea, 2008, pp. 1934–1939.
[50] C. Kirches, L. Wirsching, H. Bock, J.P. Schlöder, Efficient direct multiple shooting for nonlinear model predictive control on long horizons, J. Process Control 22 (2012) 540–550.
[51] C. Kirches, L. Wirsching, S. Sager, H.G. Bock, Efficient numerics for nonlinear model predictive control, in: M. Diehl, F. Glineur, E. Jarlebring, W. Michiels (Eds.), Recent Advances in Optimization and its Applications in Engineering, Springer, 2010, pp. 339–357.
[52] M. Diehl, R. Findeisen, F. Allgöwer, H. Bock, J.P. Schlöder, Nominal stability of real-time iteration scheme for nonlinear model predictive control, in: Control Theory and Applications, IEE Proceedings, vol. 152, 2005, pp. 296–308.
[53] M. Diehl, H.G. Bock, J.P. Schlöder, A real-time iteration scheme for nonlinear optimization in optimal feedback control, SIAM J. Control Optim. 43 (2005) 1714–1736.
[54] J. Zhao, M. Diehl, R.W. Longman, H.G. Bock, J.P. Schlöder, Nonlinear model predictive control of robots using real-time optimization, in: Proc. of the AIAA/AAS Astrodynamics Conference, Rhode Island, 2004.
[55] M. Diehl, L. Magni, G. De Nicolao, Efficient NMPC of unstable peperiod systems using approximate inifinite horizon closed loop costing, Annu. Rev. Control 28 (2004) 37–45.
[56] H.J. Ferreau, B. Houska, K. Geebelen, M. Diehl, Real-time control of a kite-model using an auto-generated nonlinear MPC algorithm, in: Proc. of the 18th IFAC World Congress, 2011, pp. 2488–2493.
[57] S. Gros, R. Quirynen, M. Diehl, Aircraft control based on fast non-linear MPC & multiple-shooting, in: Proc. of the 51st IEEE CDC, Hawaii, USA, 2012, pp. 1142–1147.
[58] C. Aurora, M. Diehl, P. Kuhl, L. Magni, R. Scattolini, Nonlinear model predictive control of combined cycle power plants, in: Proc. of the 16th IFAC World Congress, 2005, pp. 128–132.
[59] H.J. Ferreau, G. Lorini, M. Diehl, Fast nonlinear model predictive control of gasoline engines, in: Comput. Aided Control Syst. Des., 2006, pp. 2754–2759.
[60] M. Schlegel, K. Stockmann, T. Binder, W. Marquardt, Dynamic optimization using adaptive control vector parameterization, Comput. Chem. Eng. 29 (8) (2005) 1731–1751.
[61] R. Huang, V.M. Zavala, L.T. Biegler, Advanced step nonlinear model predictive control for air separation units, J. Process Control 19 (2009) 678–685.
[62] V.M. Zavala, L.T. Biegler, Optimization-based strategies for the operation of low-density polyethylene tubular reactors: nonlinear model predictive control, Comput. Chem. Eng. 33 (2009) 1735–1746.
[63] L.T. Biegler, Efficient nonlinear programming algorithms for chemical process control and operations, in: A. Korytowski, K. Malanowski, W. Mitkowski, M. Szymkat (Eds.), System Modeling and Optimization, IFIP AICT, vol. 312, Springer, Berlin, Heidelberg, 2009, pp. 21–35.
[64] V.M. Zavala, L.T. Biegler, Nonlinear programming strategies for state estimation and model predictive control, in: L. Magni, D. Raimondo, F. Allgöwer (Eds.), Nonlinear Model Predictive Control, Springer Verlag, 2009, pp. 419–432.
[65] R. Lopez-Negrete, F.J. D'Amato, L.T. Biegler, A. Kumar, Fast nonlinear model predictive control: formulation and industrial process applications, Comput. Chem. Eng. 51 (2013) 55–64.
[66] L. Grüne, J. Pannek, Analysis of unconstrained NMPC schemes with incomplete optimization, in: Proc. of NOLCOS, Bologna, Italy, 2010, pp. 238–243.
[67] K. Graichen, A. Kugi, Stability and incremental improvement of suboptimal MPC without terminal constraints, IEEE Trans. Autom. Control 55 (2010) 2576–2580.
[68] M. Diehl, R. Amrit, J. Rawlings, A Lyapunov function for economic optimizing model predictive control, IEEE Trans. Autom. Control 56 (3) (2011) 703–707.
[69] T. Backx, O. Bosgra, W. Marquardt, Integration of model predictive control and optimization of processes, in: Proc. of the Adchem, vol. 1, 2000, pp. 249–260.
[70] J. Kadam, W. Marquardt, Integration of economical optimization and control for intentionally transient process operation, in: R. Findeisen, F. Allgöwer, L. Biegler (Eds.), Assessment and Future Directions of Nonlinear Model Predictive Control, Lecture Notes in Control and Engineering Science, Springer, 2007, pp. 419–434.
[71] J. Albersmeyer, D. Beigel, C. Kirches, L. Wirsching, H.G. Bock, J.P. Schlöder, Fast nonlinear model predictive control with an application in automative engineering, in: L. Magni, D.M. Raimondo, F. Allgöwer (Eds.), Nonlinear Model Predictive Control – Towards New Challenging Applications, Lecture Notes in Control and Information Sciences, vol. 384, Springer-Verlag, 2009, pp. 471–480.
[72] L. Würth, R. Hannemann, W. Marquardt, A two-layer architecture for economically optimal process control and operation, J. Process Control 21 (3) (2011) 311–321.
[73] M. Ruben, D. Sarabia, D. Navia, C. De Prada, Coordination of distributed model predictive controllers using price-driven coordination and sensitivity analysis, in: Proc. of the DYCOPS, Mumbai, India, 2013, pp. 215–220.
[74] S. Engell, Feedback control for optimal process operation, J. Process Control 17 (2007) 203–219.
[75] W. Li, L. Biegler, Multistep, Newton-type control strategies for constrained, nonlinear processes, Chem. Eng. Res. Des. 67 (1989) 562–577.
[76] N. Oliveira, L. Biegler, An extension of Newton-type control algorithms for nonlinear process control, Automatica 31 (1995) 281–286.
[77] G. Gattu, E. Zafiriou, Nonlinear quadratic dynamic matrix control with state estimation, Ind. Eng. Chem. Res. 31 (1992) 1096–1104.
[78] Y.I. Lee, B. Kouvaritakis, M. Cannon, Constrained receding horizon predictive control for nonlinear systems, Automatica 38 (2002) 2093–2102.
[79] A.A. Tiagounov, S. Weiland, Model predictive control algorithm for nonlinear chemical processes, in: Proc. of PHYSCON 2003, 2003, pp. 334–339.
[80] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, E.F. Mishchenko, The Mathematical Theory of Optimal Processes, Interscience Publisher, 1962.
[81] T. Binder, L. Blank, H.G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schlöder, O. Stryk, Introduction to model based optimization of chemical processes on moving horizons, in: M. Grötschel, S.O. Krumke, J. Rambau (Eds.), Online Optimization of Large Scale Systems: State of the Art, Springer-Verlag, 2001, pp. 295–340 (Chapter: Moving Horizon Methods in Chemical Engineering).
[82] H.J. Pesch, Real-time computation of feedback controls for constrained optimal control problems. Part 2: A correction method based on multiple shooting, Optim. Control Appl. Met. 10 (1989) 147–171.
[83] K. Graichen, M. Egretzberger, A. Kugi, Ein suboptimal Ansatz zur schnellen modellprädiktiven Regelung, at-Automatisierungstechnik 58 (2010) 447–456.
[84] K. Graichen, A fixed-point iteration scheme for real-time model predictive control, Automatica 48 (2012) 1300–1305.
[85] K. Graichen, B. Käpernick, A real-time gradient method for nonlinear model predictive control, in: T. Zheng (Ed.), Frontiers in Model Predictive Control, InTech, 2012, pp. 9–28.
[86] K. Graichen, Nichtlineare modellprädiktive Regelung basierend auf Fixpunktiterationen, at-Automatisierungstechnik 8 (2012) 442–451.

[87] M. Kojima, Strongly stable stationary solutions in nonlinear programs, in: S.M. Robinson (Ed.), Analysis and Computation of Fixed Points, American Press, 1980, pp. 93–138.
[88] L.F. Domínguez, D.A. Narciso, E.N. Pistikopoulos, Recent advances in multiparametric nonlinear programming, Comput. Chem. Eng. 34 (2010) 707–716.
[89] A. Bemporad, M. Morari, V. Dua, E.N. Pistikopoulos, The explicit linear quadratic regulator for constrained systems, Automatica 38 (2002) 3–20.
[90] E.N. Pistikopoulos, V. Dua, N.A. Bozinis, A. Bemporad, M. Morari, On-line optimization via off-line parametric optimization tools, Comput. Chem. Eng. 24 (2000) 183–188.
[91] G. Pannocchia, J.B. Rawlings, S.J. Wright, Fast, large-scale model predictive control by partial enumeration, Automatica 43 (5) (2007) 852–860.
[92] F. Borrelli, M. Baotić, J. Pekar, G. Stewart, On the computation of linear model predictive control laws, Automatica 46 (2010) 1035–1041.
[93] M.N. Zeilinger, C.N. Jones, M. Morari, Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization, IEEE Trans. Autom. Control 56 (2011) 1524–1534.
[94] M. Jost, M. Mönnigmann, Accelerating online MPC with partial explicit information and linear storage complexity in the number of constraints, in: Proc. of the ECC, Zürich, Switzerland, 2013, pp. 35–40.
[95] A. Alessio, A. Bemporad, A survey on explicit model predictive control, in: L. Magni, D.M. Raimondo, F. Allgöwer (Eds.), Nonlinear Model Predictive Control – Towards New Challenging Applications, Lecture Notes in Control and Information Sciences, vol. 384, Springer-Verlag, 2009, pp. 345–369.
[96] L.F. Domínguez, E.N. Pistikopoulos, Recent advances in explicit multiparametric nonlinear model predictive control, Ind. Eng. Chem. Res. 50 (2011) 609–619.
[97] A. Bemporad, F. Borrelli, M. Morari, Model predictive control based on linear programming – the explicit solution, IEE Trans. Autom. Control 47 (2002) 1974–1984.
[98] T.A. Johansen, On multi-parametric nonlinear programming and explicit nonlinear model predictive control, in: Proc. of the 41st IEEE CDC, 2002, pp. 2768–2773.
[99] T.A. Johansen, Approximate explicit receding horizon control of constrained nonlinear systems, Automatica 40 (2004) 293–300.
[100] G. Pannocchia, S.J. Wright, J.B. Rawlings, Partial enumeration MPC: robust stability results and application to an unstable CSTR, J. Process Control 21 (2011) 1459–1466.
[101] T. Ohtsuka, A continuation/GMRES method for fast computation of nonlinear receding horizon control, Automatica 40 (2004) 563–574.
[102] Y. Shimizu, T. Ohtsuka, M. Diehl, A real-time algorithm for nonlinear receding horizon control using multiple shooting and continuation/krylov method, Int. J. Robust Nonlinear Control 19 (2009) 919–936.
[103] J. Marutani, T. Ohtsuka, A real-time algorithm for nonlinear receding horizon control of descriptor systems, in: Proc. of SICE Annual Conference, Taipei, Taiwan, 2010, pp. 219–222.
[104] J. Marutani, T. Ohtsuka, A real-time algorithm for nonlinear infinite horizon optimal control by time axis transformation method, Int. J. Robust Nonlinear Control 23 (2013) 1955–1971.
[105] H. Seguchi, T. Ohtsuka, Nonlinear receding horizon control of an underactuated hovercraft, Int. J. Robust Nonlinear Control 13 (2003) 381–398.
[106] D. DeHaan, M. Guay, A new real-time perspective on non-linear model predictive control, J. Process Control 16 (2006) 615–624.
[107] D. DeHaan, M. Guay, A real-time framework for model-predictive control of continuous-time nonlinear systems, IEEE Trans. Autom. Control 52 (2007) 2047–2057.