Preprints, 6th IFAC Conference on Nonlinear Model Predictive Control
Madison, WI, USA, August 19-22, 2018

A Fast NMPC Approach based on Bounded-Variable Nonlinear Least Squares

Nilay Saraf [*,**] **Mario Zanon** [**] **Alberto Bemporad** [**]

[*] *ODYS S.r.l., Via A. Passaglia 185, Lucca, 55100 LU Italy (e-mail: nilay.saraf@odys.it).*
[**] *IMT School for Advanced Studies Lucca, Piazza S. Francesco 19, Lucca, 55100 LU Italy (e-mail:{nilay.saraf,mario.zanon,alberto.bemporad}@imtlucca.it)*

Abstract: In this paper, we present an approach for real-time nonlinear model predictive control (NMPC) of constrained multivariable dynamical systems described by nonlinear difference equations. The NMPC problem is formulated by means of a quadratic penalty function as an always feasible, sparse nonlinear least-squares problem subject to box constraints on the decision variables. This formulation is exploited by the proposed fast solution algorithm, which is based on the Gauss-Newton method and bounded-variable least squares (BVLS). Linear time-invariant and linear time-varying model predictive control based on BVLS are special cases of the proposed NMPC framework. The proposed approach and its benefits are demonstrated through a typical numerical example in simulation.

Keywords: Nonlinear model predictive control, Bounded-variable least squares, Gauss-Newton method

1. INTRODUCTION

Nonlinear Model Predictive Control (NMPC) is a popular control strategy which is able to deal with constrained nonlinear systems. However, a common obstacle is the need of solving a nonconvex optimization problem within a stipulated sampling period. A common approach in tackling this issue is developing efficient algorithms tailored to NMPC problems. Often, a suboptimal solution which can be computed fast and efficiently is preferred over a precise one that requires longer computational times (Diehl et al. (2005)). Many different tools have been developed for this purpose, see e.g. (Cannon (2004); Houska et al. (2011); Stella et al. (2017); Diehl et al. (2009); Ohtsuka (2004); Li and Biegler (1989); Zavala and Biegler (2009)). In this work instead, we follow a different approach and formulate the NMPC problem in a simple way such that it becomes possible to employ existing fast optimization algorithms.

The quadratic penalty method (Nocedal and Wright (2006)) converts a constrained nonlinear optimization problem to an unconstrained one, which is solved iteratively by incrementing the penalty parameter until the solution is achieved. Using a large enough value of the penalty parameter yields an approximately optimal solution in a single iteration. The need of adequately selecting the penalty parameter to balance accuracy of the solution and ill-conditioning of the problem made this approach not the most appealing for general-purpose solvers. In NMPC, however, small constraints violations are typically negligible compared to model inaccuracy and external perturbations acting on the system. Therefore, the quadratic penalty method is very appealing for such applications due to the possibility of developing efficient implementations. We propose to use a single iteration of the quadratic penalty method which keeps simple bounds on the decision variables as such and relaxes the equality constraints via a quadratic penalty function. The obtained problem is bound-constrained nonlinear least-squares (NLLS), which we solve by employing the Gauss-Newton method (Björck (1996)) and a Bounded-Variable Least-Squares (BVLS) solver, which is both efficient and numerically robust.

As opposed to standard infeasibility handling approaches which relax the output constraints (Scokaert and Rawlings (1999); Houska et al. (2011)), we relax the equality constraints related to the model of the system. The main motivations for such a choice are that (a) it preserves feasibility of the optimization problem, similarly to the output constraint relaxation, and (b) the available model is an approximate representation of the true system. In particular, we show through a typical numerical example that the constraint violation is not significant unless the original problem becomes infeasible, and therefore, the control performance is not deteriorated.

We first define in Section 2 the class of models, perfomance index, and constraints that are typically considered for formulating NMPC problems. Based on that, the NMPC problem formulations which we consider are described in Section 3, where we also derive the proposed NMPC formulation. Next, in Section 4 we describe in detail the optimization algorithm which we employ in order to solve the resulting nonlinear optimization problem. Numerical results and their discussion are presented in Section 5 with concluding remarks in Section 6.

[⋆] This work has been supported by the European Commission under the Horizon 2020 programme, grant agreement no. 674875 (oCPS)

2. PRELIMINARIES

We describe the system dynamics by using the following input-output model

$$f(Y_k, U_k, V_k) = \mathbf{0}, \tag{1}$$

where we define the inputs and outputs respectively as $U_k = (u_{k-n_\mathrm{b}}, \dots, u_{k-1})$, $u_k \in \mathbb{R}^{n_u}$, $Y_k = (y_{k-n_\mathrm{a}}, \dots, y_k)$, $y_k \in \mathbb{R}^{n_y}$. Vector $V_k = (v_{k-n_\mathrm{c}}, \dots, v_{k-1})$, $v_k \in \mathbb{R}^{n_v}$ defines a measured disturbance and $\mathbf{0}$ represents a zero vector. Function $f : \mathbb{R}^{n_\mathrm{a} n_y} \times \mathbb{R}^{n_\mathrm{b} n_u} \times \mathbb{R}^{n_\mathrm{c} n_v} \to \mathbb{R}^{n_y}$ is in general nonlinear where $n_\mathrm{a}, n_\mathrm{b}$ and n_c define the model order. We assume it to be differentiable, i.e. of class C^1. The class of nonlinear models of the form (1) includes for instance, state-space models and the noise-free polynomial NARX (nonlinear autoregressive exogenous) models (Leontaritis and Billings (1985)).

We consider a convex quadratic performance index 'J', which is separable in time and a typical choice for regulation and reference tracking in MPC:

$$\mathrm{J}(k) = \sum_{j=1}^{N_\mathrm{p}} \frac{1}{2} \| W_y^{\frac{1}{2}} (y_{k+j} - \bar{y}_{k+j}) \|_2^2 + \sum_{j=0}^{N_\mathrm{u}-2} \frac{1}{2} \| W_u^{\frac{1}{2}} (u_{k+j} - \bar{u}_{k+j}) \|_2^2$$
$$+ \frac{1}{2} (N_\mathrm{p} - N_\mathrm{u} + 1) \cdot \| W_u^{\frac{1}{2}} (u_{k+N_\mathrm{u}-1} - \bar{u}_{k+N_\mathrm{u}-1}) \|_2^2, \tag{2}$$

where N_p and N_u denote the prediction and control horizon respectively. Matrices $W_y \in \mathbb{R}^{n_y \times n_y}$ and $W_u \in \mathbb{R}^{n_u \times n_u}$ are positive semidefinite tuning weights, and $\bar{y}$, $\bar{u}$ denote the references for outputs and inputs, respectively. In general, the proposed NMPC approach described in Section 3.3 is not limited to the above mentioned performance index. Depending on the control objectives, any cost function that results in a sum of squares of linear or C^1 nonlinear functions may be considered in order to formulate the NMPC problem.

As we will explain in Section 4, in order to exploit an efficient solution algorithm, the only inequality constraints that we consider are simple bounds on the optimization variables. General inequalities (3) can be converted to the considered setting by introducing slack variables $\nu \in \mathbb{R}^{n_\mathrm{i}}$ having non-negativity constraints such that

$$\begin{aligned} g(w_k) \leq \mathbf{0} \quad &\text{becomes,} \\ g(w_k) + \nu_k = \mathbf{0} \quad &\text{and} \quad \nu_k \geq 0, \end{aligned} \tag{3}$$

where $w_k = (u_k, \dots, u_{k+N_\mathrm{u}-1}, y_{k+1}, \dots, y_{k+N_\mathrm{p}})$, $z_k = (w_k, \nu_k)$ and $g : \mathbb{R}^{(n_z - n_\mathrm{i})} \to \mathbb{R}^{n_\mathrm{i}}$ is assumed to be differentiable of class C^1, while n_z and n_i denote the number of decision variables and general inequality constraints respectively. We summarise all NMPC constraints at each time step k as

$$h(z_k) = \mathbf{0} \tag{4}$$
$$p_k \leq z_k \leq q_k \tag{5}$$

where $p_k, q_k \in \mathbb{R}^{n_z}$ are vectors defining bounds on the input and output variables, and non-negativity constraint on the slack variables. Owing to the construction of equalities (4), it is worth noting that the Jacobian matrix of h w.r.t. z evaluated at any given z_k is sparse, with its sparsity pattern depending on the chosen model (1).

3. NMPC PROBLEM FORMULATIONS

3.1 Constrained NLP

Employing the performance index (2) and the constraint set defined by (4) and (5), the NMPC problem can be defined as the following constrained NLP

$$\min_{z_k} \frac{1}{2} \| W(z_k - \bar{z}) \|_2^2 \tag{6a}$$
$$\text{s.t. } h(z_k) = 0, \tag{6b}$$
$$p_k \leq z_k \leq q_k, \tag{6c}$$

where, $W \in \mathbb{R}^{(n_z - n_\mathrm{i}) \times n_\mathrm{i}}$ is block-sparse and is constructed from square root of the tuning weights (W_y, W_u) defined in (2), $\bar{z}$ is constructed from the input and output references such that the elements corresponding to slack variables are zero. While one usually avoids the use of slack variables in the definition of the general (nonlinear) inequalities (3), we abide by this definition, since it is required by the formulation proposed in Section 3.3.

3.2 Soft-constrained NLP

In practice, due to unmodeled dynamics and perturbations acting on the system, the imposed constraints might become impossible to satisfy such that Problem (6) becomes infeasible. In order to preserve feasibility, a common approach is to introduce additional slack variable(s) ϵ in the problem to relax the output constraints, which then become *soft constraints*. Details regarding this approach based on exact penalty functions may be referred in (Borrelli et al. (2017); Kerrigan and Maciejowski (2000)). For completeness, we provide a possible relaxation of (6) using an exact penalty approach.

$$\min_{z_k, \epsilon} \frac{1}{2} \| W(z_k - \bar{z}) \|_2^2 + \frac{\sigma_1}{2} \| \epsilon \|_2^2 + \sigma_2 \epsilon \tag{7a}$$
$$\text{s.t. } h(z_k) = 0, \tag{7b}$$
$$p_k - V\epsilon \leq z_k \leq q_k + V\epsilon, \tag{7c}$$
$$\epsilon \geq 0, \tag{7d}$$

where V is a matrix with all elements non-negative such that only the output constraints are relaxed and the remaining constraints are strictly satisfied. The tuning weights σ_1, σ_2 are such that $\sigma_1 \geq 0$ is a small weight, and $\sigma_2 > 0$ is a large weight which ensures that $\epsilon > 0$ only when Problem (6) becomes infeasible and $\epsilon = 0$ otherwise. Problem (7), which we will refer to as 'NLP-soft' can be solved by an NLP solver. In this paper, however, we propose to use a different reformulation of (6) which allows us to use algorithms based on ideas recently proposed in Saraf and Bemporad (2017).

3.3 Bound-constrained NLLS

The quadratic penalty method (Nocedal and Wright, 2006, Sec. 17.1), in just one iteration, with a large value of penalty ρ yields a solution close to a local or global minimum which is usually sufficient for MPC problems. Based on this, instead of relaxing the output bounds, we propose to relax the equality constraints and penalize their violation with a quadratic penalty as

$$\min_{p_k \leq z_k \leq q_k} \frac{1}{2} \| W(z_k - \bar{z}) \|_2^2 + \frac{\rho}{2} \| h(z_k) \|_2^2$$

where $\rho > 0$ is a large weight. Alternatively,

$$\min_{p_k \le z(k) \le q_k} \frac{1}{2} \left\| \begin{matrix} \frac{1}{\sqrt{\rho}} W(z_k - \bar{z}) \\ h(z_k) \end{matrix} \right\|_2^2 \equiv \min_{p_k \le z(k) \le q_k} \frac{1}{2} \|r(z_k)\|_2^2, \tag{8}$$

where, $r : \mathbb{R}^{n_z} \to \mathbb{R}^{n_r}$ is implicitly defined by the above equivalence and is referred to as the vector of residuals. While it is desirable to have a large value of penalty ρ such that the introduced inaccuracy is minimal, in order to avoid numerical issues due to ill-conditioning of the problem, ρ must not be too large. Note that Problem (8) which we will refer to as 'NLLS-box', cannot become infeasible as the equality constraints are relaxed. Relaxing the equality constraints has a practical justification as the prediction model (1) is only an approximation of the actual system dynamics and rather opportunistic violations of these equations only act as a perturbation in the predictions, or in other words, change the magnitude of model mismatch. In Section 5, we will show that the constraint violations are significant only when problem (6) becomes infeasible.

4. OPTIMIZATION ALGORITHM

Problem (8) can be efficiently solved using the bounded-variable nonlinear least-squares (BVNLLS) algorithm described in Algorithm 1. The main idea is to solve the NLLS-box problem by an approach that follows the steps of a Gauss-Newton Hessian approximation based Sequential Quadratic Programming (SQP) (Nocedal and Wright (2006)) while exploiting the special structure of (8). The Gauss-Newton method (Björck, 1996, Sec. 9.2) is a well known approach to solve Nonlinear Least Squares (NLLS) problems and is based on a sequence of linear approximations of the residual function $r(z)$ in (8). BVNLLS is an extension of the Gauss-Newton method to handle box-constraints, in which a box-constrained Least Squares (LS-box) problem is solved in each iteration until termination criteria are met. If z_i denotes the current solution estimate, then a correction step Δz is computed as a solution to the LS-box problem

$$\min_{p - z_i \le \Delta z \le q - z_i} \|J(z_i)\Delta z + r(z_i)\|_2^2, \tag{9}$$

where p and q denote the bounds on z, $J(z) = \nabla_z r(z)^\top$ is the Jacobian matrix of $r(z)$ w.r.t. z, and the new solution estimate is $z_{i+1} = z_i + \Delta z$. We solve Problem (9) using an efficient implementation of the bounded-variable least-squares (BVLS) algorithm (Stark and Parker (1995)), which is a primal active-set method and we initialize it with a zero vector as the initial guess. Convergence of the BVNLLS algorithm that involves Gauss-Newton step computations is ensured by including a backtracking line-search method (Nocedal and Wright, 2006, Sec. 3.1) based on the Armijo-Goldstein condition (Björck, 1996, Sec. 9.2.1), which forms Steps 6-10 of the BVNLLS algorithm. Necessary and sufficient conditions for the termination of the algorithm are described as follows based on first-order optimality conditions that must be satisfied by a local or global minimum of (8). We first note that Algorithm 1 starts with an initial guess (coldstart or a warmstart) that satisfies the bounds. Moreover, since Step 5 of Algorithm 1 generates a step such that the next iterate also satisfies the bounds, the primal feasibility condition of problem (8) is satisfied at any iteration. Let us denote the Lagrange multipliers for lower and upper bounds respectively as λ_l and λ_u, and the sets of active lower and upper bounds respectively as $\mathcal{L}$ and $\mathcal{U}$. We denote the gradient of the NLLS cost function at iterate i as

Algorithm 1 Bounded-Variable Nonlinear Least Squares (BVNLLS)

Inputs: Vectors p and q that define bounds on z, initial guess $z \in \{z_0 | p \le z_0 \le q\}$, $b = r(z)$, optimality tolerance $\gamma \ge 0$, $c \in (0,1)$ and $\tau \in (0,1)$;

```
1:  $J \leftarrow \mathcal{J}(z)$, $\mathcal{J}_{ij} = \frac{\partial r_i}{\partial z_j}, i \in \{1, \cdots, n_r\}, j \in \{1, \cdots, n_z\}$;
2:  $\mathcal{L} \leftarrow \{j | z(j) \le p(j)\}$; $\mathcal{U} \leftarrow \{j | z(j) \ge q(j)\}$;
3:  $d \leftarrow J^\top b$;
    $\lambda_l(j) \leftarrow d(j), \forall j \in \mathcal{L}$; $\lambda_u(j) \leftarrow -d(j), \forall j \in \mathcal{U}$;
4:  if $\lambda_l(j) \ge \gamma, \forall j \in \mathcal{L}$ and $\lambda_u(j) \ge \gamma, \forall j \in \mathcal{U}$ and
    $|d(j)| \le \gamma, \forall j \notin \mathcal{L} \cup \mathcal{U}$ then go to Step 12;
5:  $\Delta z \leftarrow \arg \min_{p - z \le \Delta z \le q - z} \|J\Delta z + b\|_2^2$
6:  $\alpha = 1$; $\theta \leftarrow c\alpha\|J\Delta z\|_2^2$; $\psi \leftarrow b^\top b$; $b \leftarrow r(z + \Delta z)$;
    $\phi \leftarrow b^\top b$;
7:  while $\phi > \psi - \theta$ do
8:  $\alpha \leftarrow \tau\alpha$; $\theta \leftarrow \alpha\theta$;
9:  $b \leftarrow r(z + \alpha\Delta z)$; $\phi \leftarrow b^\top b$;
10: end while
11: $z \leftarrow z + \alpha\Delta z$; go to Step 1;
12: $z^\star \leftarrow z$; $\lambda_l(j) \leftarrow 0, \forall j \notin \mathcal{L}$; $\lambda_u(j) \leftarrow 0, \forall j \notin \mathcal{U}$;
13: end.
```

Outputs: Local or global optimum $z^\star$ of (8), objective function value at $z^\star = \phi$, and Lagrange multiplier vectors λ_l and λ_u corresponding to lower and upper bounds respectively.

$$d_i := J^\top(z_i) r(z_i).$$

If z_i is a local minimum of (8), then the following KKT conditions hold:

$$d_i - \lambda_l + \lambda_u = \mathbf{0}, \tag{10a}$$
$$\lambda_l(j) = d_i(j), \quad \forall j \in \mathcal{L}, \tag{10b}$$
$$\lambda_u(j) = -d_i(j), \quad \forall j \in \mathcal{U}, \tag{10c}$$
$$d_i(j) = \mathbf{0}, \quad \forall j \in \{[1, n_z]\} \setminus \{\mathcal{L} \cup \mathcal{U}\}, \tag{10d}$$
$$d_i(j) \ge \mathbf{0}, \quad \forall j \in \mathcal{L}, \tag{10e}$$
$$d_i(j) \le \mathbf{0}, \quad \forall j \in \mathcal{U}. \tag{10f}$$

where we used $\mathcal{L} \cap \mathcal{U} = \emptyset$. This defines the termination criterion evaluated in Step 4 of Algorithm 1.

The LS-box Problem (9) only has simple bounds, which allows the corresponding convex QP to be solved by a primal active-set method (Nocedal and Wright, 2006) by applying efficient linear algebra routines to the smaller-dimensional space of the primal variables which are not at their bounds. For generic QPs, instead, one needs to work in a higher dimensional space which also includes the Lagrange multipliers. In BVLS, the Lagrange multipliers corresponding to the active-set are obtained simply from respective entries in the gradient vector of the least-squares cost. The same characteristic is also present in BVNLLS, implied by (10) and Step 3 of Algorithm 1. Moreover, for generic QPs, active-set methods require to form and factorize the Hessian matrix $J(z_i)^\top J(z_i)$, which has a condition number squared as compared to that of the Jacobian matrix $J(z_i)$. In our setting, we expect to have a high condition number due to the necessity of choosing the penalty parameter ρ large enough. This can cause ill-conditioning while solving QPs based on the Hessian matrix even when it does not occur in BVLS, in which the matrix that is factorized is the Jacobian. A detailed description and comparison of different QP solvers for LS-box is beyond the scope of this paper and will be the subject of future publications.

In summary, the above mentioned characteristics make BVNLLS computationally efficient, and comparatively numerically robust. It is also interesting to note that a single full Gauss-Newton step of Algorithm 1 would generate a solution that is equivalent to the one produced by the Real-Time Iteration (RTI) scheme (Diehl et al. (2005)), a special case of linear time-varying MPC (Gros et al. (2016)). This solution can then be used to provide a warmstart for the next time instant (cf. Diehl et al. (2009); Gros et al. (2016)).

Relation Between BVNLLS and SQP A popular method to solve the NLP (6) is the sequential quadratic programming (SQP) algorithm. The QP subproblem solved in each Gauss-Newton Hessian approximation based SQP iteration i is

$$\begin{aligned} &\min_{\Delta z} \|J_i \Delta z + b_i\|_2^2 \\ &\text{s.t. } \nabla h(z_i)^\top \Delta z + h(z_i) = \mathbf{0}, \\ &\quad p - z_i \le \Delta z \le q - z_i. \end{aligned} \tag{11}$$

A quadratic penalty function based elimination of the linear equality constraints (11) would result in the same LS-box problem that is solved in Step 5 of the BVNLLS algorithm. Hence, BVNLLS is similar to a Gauss-Newton SQP algorithm which relaxes the equality constraints and penalizes the square of their violation.

There are two main differences between BVNLLS and the LS-box based SQP formulation. The first one is related to how the Lagrange multipliers of the box constraints are computed: while in SQP they are derived from the solution of the QP subproblem, in BVNLLS the structure of the problem is exploited to directly compute them from Step 3 of Algorithm 1. The second one is related to the merit function used in the linesearch procedure. Since there are no equality constraints involved in BVNLLS iterations, the merit function is the cost function itself, whereas in the case of SQP, the following ℓ_1 merit function is used

$$\phi(z_i, \mu) = \|W(z_i - \bar{z})\|_2^2 + \mu \|h(z_i)\|_1$$

where, $\mu > \|\lambda_{i+1}\|_\infty$ is a sufficiently large penalty parameter (Nocedal and Wright, 2006, Sec. 18.1). As a result, while the step is computed in the same way, the LS-box based SQP is not guaranteed to always find a descent direction, since $h(z) = 0$ is never enforced exactly. This is not a problem in BVNLLS, as the merit function and the step computation are consistent.

5. NUMERICAL EXAMPLE AND RESULTS

In this section, the proposed NMPC formulation based on BVNLLS optimization is tested for performance in simulations. It is compared against the performance of benchmark NLP formulations (6) and (7) in terms of quality of control and execution speed.

5.1 Simulation setup

For the purpose of illustration, we consider the Continuous Stirred Tank Reactor (CSTR) (Seborg et al. (2004)), which is common in the process industry and is an open-loop unstable system with highly nonlinear dynamics. All simulations have been performed in MATLAB R2015b [1] using the continuous-time model of the CSTR system present in the model predictive control toolbox as a reference for the true system. Discretizing the model equations using the forward Euler method with a sampling period $t_s = 6$ seconds and substituting the model parameters, the following nonlinear discrete-time prediction model of the form (1) is obtained which represents the continuous-time model accurately enough:

$$T_{k+1} = T_k + t_s(T_{f_k} - 1.3T_k + \kappa_1 C_{A_k} e^{\frac{-5963.6}{T_k}} + 0.3T_{j_k}), \tag{12a}$$

$$C_{A_{k+1}} = C_{A_k} + t_s(C_{Af_k} - \kappa_2 C_{A_k} e^{\frac{-5963.6}{T_k}} - C_{A_k}), \tag{12b}$$

where constants $\kappa_1 = 416375136, \kappa_2 = 34930800$; T, T_j and T_f respectively denote the temperatures of the reactor, jacket coolant and the feedstream in K; C_A and C_{Af} respectively denote the concentration of reagent in the reactor and the feedstream in kgmol/m^3. The control objective here is to manipulate the input T_j in order to keep C_A at the desired set-point in the presence of measured disturbances T_f and C_{Af}. The simulation setup here is kept similar to the CSTR demo in the MPC toolbox of MATLAB. For testing the controller in reference tracking of the concentration C_A, a ramp reference as shown in Figure 1 is considered. The corresponding temperature references for the jacket coolant and reactor are computed from (12) by assuming the measured disturbances to be constant in prediction. In order to test the controller's performance in regulation, the feedstream temperature is fluctuated as a sinusoid with white measurement noise acting on all temperature states. The constraints [2] that must be satisfied are upper and lower bounds on the variables and input rate. In order to handle the input rate constraints, the bounds on the first input increment in predictions are imposed. This can be done by replacing the lower bound $u_{k_{\min}}$ by $\max\{u_{k_{\min}}, u_{k-1} + \Delta u_{\min}\}$ and the upper bound $u_{k_{\max}}$ by $\min\{u_{k_{\max}}, u_{k-1} + \Delta u_{\max}\}$ where u represents the input, Δu its increment with limits indicated by the subscript.

The MPC problems are formulated as described in Section 3. BVNLLS uses a MATLAB-interfaced C implementation of the BVLS algorithm based on recursive QR updates. The NLP problems (6), (7) are solved using MATLAB's 'fmincon' solver with SQP and for comparisons, also the sparse NLP solver 'IPOPT' (Wächter and Biegler (2006)) compiled in MATLAB with the 'MA57' linear systems solver. The solver IPOPT is considered

[1] The codes have been run on a Macbook Pro 2.6 GHz Intel Core i5 with 8GB RAM.

[2] The reactor's temperature is constrained in the range of 300-400 K whereas its concentration lies within 0-10 kgmol/m^3. The input is constrained within the range of 240-360 K and its rate is limited within ± 2 K/min.

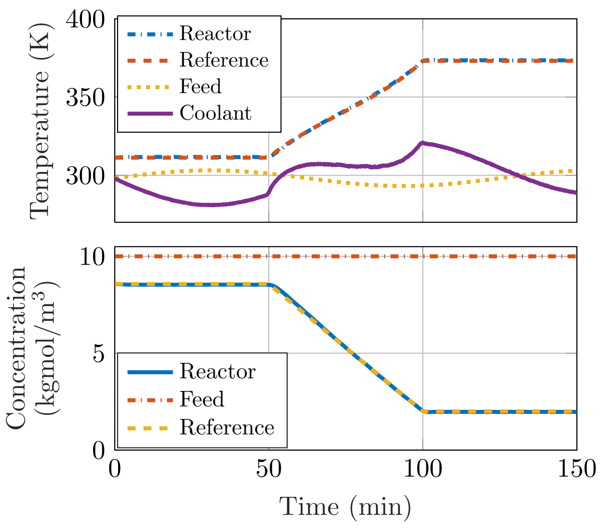

Fig. 1. Performance in tracking and regulation using formulation (8) and BVNLLS solver.

with manually supplied exact sparse Jacobian evaluation functions including sparsity pattern, and an initial guess for the primal and dual variables in order to incorporate all benefits of the tool. The fmincon SQP solver of MATLAB is also provided with gradient evaluation functions and a warmstart for faster convergence.

5.2 *Control performance comparison*

Figure 1 demonstrates the quality of performance in reference tracking and regulation of the proposed NMPC approach. The prediction horizon is set to 10 steps and the control horizon to 1 step. The weight on the coolant's and reactor's temperature tracking error is set to 1 and 10 respectively, whereas the weight on reactor concentration's tracking error is set to 100. The value of the penalty parameter ($\sqrt{\rho}$) in (8) is set to 10^4. The solvers are initialized with the feasible initial guess $z = 0.5(p + q)$ and future warmstarts are derived by using the shift-initialization technique (Diehl et al. (2005); Gros et al. (2016)). We compare the cost function values achieved with the considered formulations in Figure 2. We also observe that the equality constraints are almost strictly satisfied by the proposed NMPC approach even though they were relaxed with a quadratic penalty function. In conclusion, the same quality of performance is achieved as compared to the NLP formulation (6) due to a negligible difference between the optimal value of the cost functions achieved in the two cases.

5.3 *Execution time comparison*

Figure 3 shows that the BVNLLS solver is considerably faster for small to medium sized problems (30 to 150 decision variables and same number of bilateral bound constraints). Note that the BVNLLS and fmincon solvers have a numerically dense implementation whereas the IPOPT solver uses sparse numerical methods. Observing these comparisons demonstrate that the BVNLLS based approach is an attractive practical alternative to solve NMPC problems in real-time embedded applications where computational complexity may restrict the use of MPC.

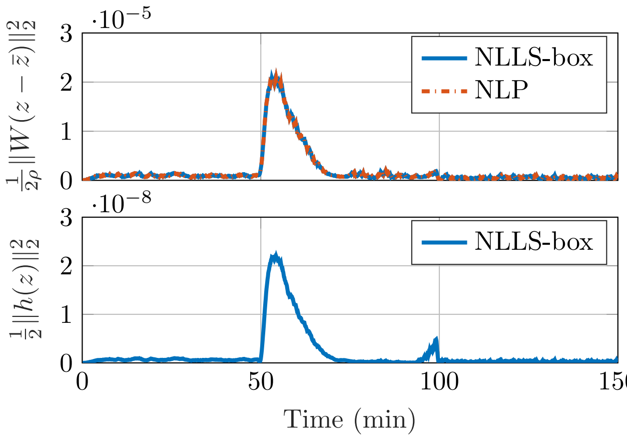

Fig. 2. Comparison of the optimized cost for the problem formulations (6), (8).

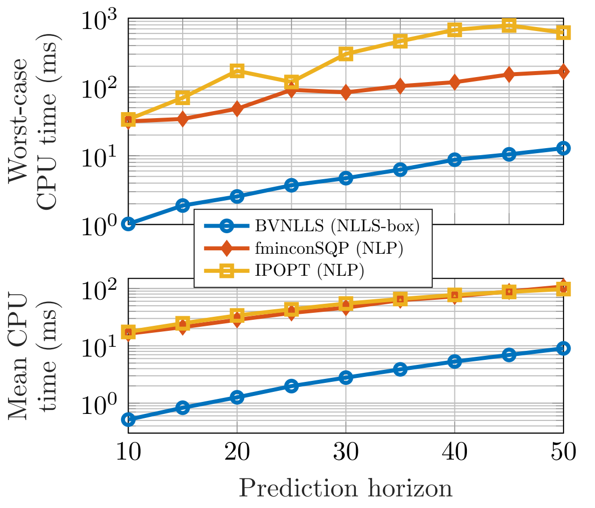

Fig. 3. CPU time for each solver during closed-loop simulation of the CSTR w.r.t. prediction horizon ($N_p = N_u$, number of variables and box constraints = $3N_\text{p}$, number of equality constraints = $2N_\text{p}$).

5.4 *Infeasibility handling performance comparison*

Finally, the way in which infeasibility is handled by the formulations (7) and (8) is compared through a simulation scenario with the CSTR system, such that the reactor temperature reference (373.13 K) exceeds the reduced upper bound (370 K). As shown in Figure 4, given the short prediction horizon ($N_p = 10, N_u = 1$), formulation (6) becomes inapplicable as the constraints become too strict to satisfy when the reactor temperature almost reaches its limit and the coolant temperature cannot reach fast enough in time to a value that would satisfy the constraints. The slack variable ϵ in (7) is only active when (6) is infeasible (cf. Figure 5). Comparing this against the equality constraints' relaxation in the NLLS-box case, it is clear that the constraint violations occur noticeably and essentially so, only when the NLP problem (6) becomes infeasible. Figure 4 shows that the trajectories in both cases coincide due to actuator saturation, which leads to the same inputs computed during instances of infeasibility. In general, one might expect different trajectories if the infeasibilities arise when the actuator limits are far. This fact is observed in the bottom part of Figure 5 where we compare the value of the performance index, which is a

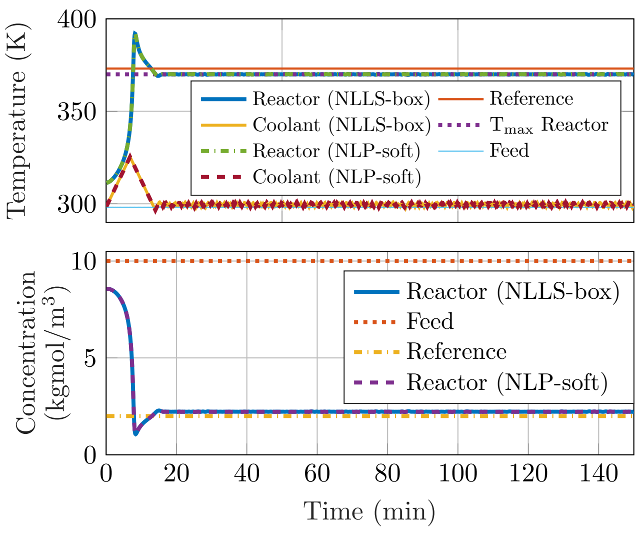

Fig. 4. Closed-loop trajectories using the NLLS-box and soft-constrained NLP approaches.

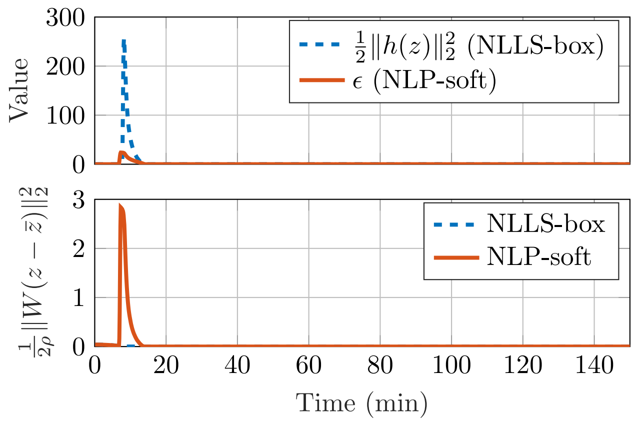

Fig. 5. Constraint relaxation comparison between the NLLS-box and soft-constrained NLP approaches.

function of the predicted sequence of inputs and outputs. The way infeasibility is handled in both cases is similar to the linear MPC case, which is thoroughly discussed in Saraf and Bemporad (2017).

6. CONCLUSION

In this paper an NMPC approach based on a simple box-constrained nonlinear least-squares formulation with a fast solution method has been proposed. Results suggest that the approach can be an appealing alternative in practical applications where fast computations are a priority. Future work includes investigation of theoretical conditions on the penalty parameter such that closed-loop stability properties are preserved, an issue that has already been solved for the linear MPC case (Saraf and Bemporad (2017)).

REFERENCES

Björck, A. (1996). *Numerical Methods for Least Squares Problems.* Society for Industrial and Applied Mathematics.

Borrelli, F., Bemporad, A., and Morari, M. (2017). *Predictive control for linear and hybrid systems.* Cambridge University Press.

Cannon, M. (2004). Efficient nonlinear model predictive control algorithms. *Annual Reviews in Control*, 28(2), 229–237.

Diehl, M., Bock, H., and Schlöder, J. (2005). A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control. *SIAM Journal on Control and Optimization*, 43(5), 1714–1736.

Diehl, M., Ferreau, H., and Haverbeke, N. (2009). Efficient Numerical Methods for Nonlinear MPC and Moving Horizon Estimation. In L. Magni, D. Raimondo, and F. Allgöwer (eds.), *Nonlinear Model Predictive Control. Lecture Notes in Control and Information Sciences*, volume 384, 56–98. Springer, Berlin, Heidelberg.

Gros, S., Zanon, M., Quirynen, R., Bemporad, A., and Diehl, M. (2016). From linear to nonlinear MPC: bridging the gap via the real-time iteration. *International Journal of Control*, 1–19.

Houska, B., Ferreau, H., and Diehl, M. (2011). ACADO toolkit - An opensource framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3), 298–312.

Kerrigan, E. and Maciejowski, J. (2000). Soft constraints and exact penalty functions in model predictive control. In *Proc. UKACC International Conference (Control).* Cambridge, UK.

Leontaritis, I. and Billings, S. (1985). Input-output parametric models for non-linear systems. Part i: deterministic non-linear systems. *International Journal of Control*, 41, 303–328.

Li, W. and Biegler, L. (1989). A Multistep, Newton-type control strategy for constrained, nonlinear processes. In *1989 American Control Conference*, 1526–1527. Pittsburgh, PA, USA.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization.* 2nd ed. Springer.

Ohtsuka, T. (2004). A continuation/gmres method for fast computation of nonlinear receding horizon control. *Automatica*, 40(4), 563–574.

Saraf, N. and Bemporad, A. (2017). Fast model predictive control based on linear input/output models and bounded-variable least squares. In *Proc. 56th IEEE Conference on Decision and Control*, 1919–1924. Melbourne, Australia.

Scokaert, P. and Rawlings, J. (1999). Feasibility issues in linear model predictive control. *AIChE J.*, 45(8), 1649–1659.

Seborg, D., Edgar, T.F., and Mellichamp, D.A. (2004). *Process Dynamics and Control.* 2nd ed. Wiley.

Stark, P. and Parker, R. (1995). Bounded-Variable Least-Squares: An Algorithm and Applications. *Computational Statistics*, 10, 129–141.

Stella, L., Themelis, A., Sopasakis, P., and Patrinos, P. (2017). A Simple and Efficient Algorithm for Nonlinear Model Predictive Control. In *Proc. 56th IEEE Conference on Decision and Control*, 1939–1944. Melbourne, Australia.

Wächter, A. and Biegler, L. (2006). On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming. *Mathematical Programming*, 106(1), 25–57.

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