A Simple and Efficient Algorithm for Nonlinear Model Predictive Control

Lorenzo Stella, Andreas Themelis, Pantelis Sopasakis and Panagiotis Patrinos

Abstract— **We present `PANOC`, a new algorithm for solving optimal control problems arising in nonlinear model predictive control (NMPC). A usual approach to this type of problems is sequential quadratic programming (SQP), which requires the solution of a quadratic program at every iteration and, consequently, inner iterative procedures. As a result, when the problem is ill-conditioned or the prediction horizon is large, each outer iteration becomes computationally very expensive. We propose a line-search algorithm that combines forward-backward iterations (FB) and Newton-type steps over the recently introduced forward-backward envelope (FBE), a continuous, real-valued, exact merit function for the original problem. The curvature information of Newton-type methods enables asymptotic superlinear rates under mild assumptions at the limit point, and the proposed algorithm is based on very simple operations: access to first-order information of the cost and dynamics and low-cost direct linear algebra. No inner iterative procedure nor Hessian evaluation is required, making our approach computationally simpler than SQP methods. The low-memory requirements and simple implementation make our method particularly suited for embedded NMPC applications.**

I. Introduction

Model predictive control (MPC) has become a popular strategy to implement feedback control loops for a variety of systems, due to its ability to take into account for constraints on inputs, states and outputs. Its success is intimately tied to the availability of efficient, reliable algorithms for the solution of the underlying constrained optimization problem: linear MPC requires solving a convex QP at every sampling step, for which the mature theory of convex optimization provides simple and robust methods with global convergence guarantees.

On the other hand, the vast majority of systems are nonlinear by nature, and nonlinear models often capture their dynamics much more accurately. For this reason nonlinear MPC (NMPC) is a well suited approach to design feedback controllers in many cases. At every sampling step, NMPC requires the solution of a general nonlinear program (NLP): general approaches for NLP include sequential quadratic programming (SQP) and interior-point methods (IP) [1], [2]. Typically this NLP represents a discrete-time approximation of the continuous-time, and thus infinite-dimensional, constrained nonlinear optimal control problem, within a direct optimal control framework. Various ways exist for deriving a finite-dimensional NLP from a continuous-time optimal control problem, namely single shooting, multiple shooting and collocation methods, see e.g. [3], [4]. Although multiple shooting formulations (keeping the states as problem variables) are recently popular, single shooting formulations (implicitly eliminating the states) have traditionally been used to exploit the sequential structure in optimal control problems, see [5], [6] and [2, §2.6] for a textbook account.

The authors are with the Department of Electrical Engineering (ESAT-STADIUS) and Optimization in Engineering Center (OPTEC), KU Leuven, Kasteelpark Arenberg 10, 3001 Leuven, Belgium. The first two authors are also affiliated with the IMT School for Advanced Studies Lucca, Piazza S. Francesco 17, 55100 Lucca, Italy.

A. Problems framework and motivation

In this paper we deal with discrete-time, optimal control problems with nonlinear dynamics. This type of problems can be obtained, for example, by appropriately discretizing continuous-time problems. Furthermore, we allow for nonsmooth (possibly nonconvex) penalties on the inputs: these can be (hard or soft) input constraints, or could be used for example to impose (group) sparsity on the input variables by using sparsity-inducing penalties. Note that problems with soft state constraints fit this framework by including an additional smooth penalty on the system state (*e.g.*, the squared Euclidean distance from a constrained set), in the spirit of a generalized quadratic penalty method.

By eliminating the state variables and expressing the cost as a function of the inputs only (single-shooting formulation), the NMPC problems that we address can be reduced to the minimization of a smooth, nonconvex function f plus a nonsmooth (possibly nonconvex) penalty g. This is precisely the form of problems that can be solved by the proximal gradient method, also known as forward-backward splitting (FBS), see [7], a generalization of the projected gradient method. FBS is a fixed-point iteration for solving a nonsmooth, nonlinear system of equations defining the stationary points of the cost function. As such, its iterations are very simple and ideal for embedded applications. However, the simplicity of FBS comes at the cost of slow convergence to stationary points. In fact, like all first-order methods, the behaviour of FBS is greatly affected by the problem conditioning: in the case of NMPC, it is customary to have ill-conditioned problems due to the nonlinear dynamics and the horizon length.

B. Contributions and related works

We propose a new, simple method for solving NMPC problems. The proposed algorithm is a line-search method for solving the fixed-point equations associated with FBS, using the so-called *forward-backward envelope* (FBE) as merit function to determine the stepsize [8], [9], [10]. We show that if the search directions are computed using quasi-Newton formulas, then the algorithm converges with superliner asymptotic rate to a stationary point. Computing the directions and evaluating the FBE simply require the computation of

the forward-backward mapping, therefore the proposed algorithm is based on the very same operations as FBS:

1) evaluation of the gradient of the smooth cost, which is performed using automatic differentiation (AD);
2) evaluation of the proximal mapping of the nonsmooth penalty, which usually has a very simple closed-form.

In particular, no second-order information on the problem cost is required. Toolboxes for AD that use code generation to evaluate gradients and Jacobians efficiently, such as CasADi [11], are available. Furthermore, limited-memory methods such as L-BFGS [12] that only perform inner products can be used to determine line-search directions, making the algorithm completely matrix-free and well suited for embedded implementations and applications.

A similar approach was recently exploited to analyze and accelerate the convergence of another proximal splitting algorithm in nonconvex settings, namely the Douglas-Rachford splitting, and its dual counterpart ADMM [13].

The paper is organized as follows: in §II we frame the family of problems which we target; in §III we describe the proposed method and discuss its properties; §IV shows numerical results with the proposed algorithm.

II. Problem Formulation

We consider the following finite-horizon problem

$$\begin{aligned} \textbf{minimize} \quad & \textstyle\sum_{n=0}^{N-1} \ell_n(x_n, u_n) + g_n(u_n) + \ell_N(x_N) && \text{(1a)} \\ \textbf{subject to} \quad & x_0 = \bar{x} && \text{(1b)} \\ & x_{n+1} = f_n(x_n, u_n), \ n = 0, \dots, N-1 && \text{(1c)} \end{aligned}$$

where $f_n : \mathbb{R}^{n_x \times n_u} \to \mathbb{R}^{n_x}$, $n = 0, \dots, N-1$ are smooth mappings representing system dynamics $\ell_n : \mathbb{R}^{n_x \times n_u} \to \mathbb{R}$, $n = 0, \dots, N-1$, and $\ell_N : \mathbb{R}^{n_x} \to \mathbb{R}$ are smooth functions representing stage and terminal costs respectively, and $g_n : \mathbb{R}^{n_u} \to \overline{\mathbb{R}}$, $n = 0, \dots, N-1$, are possibly *nonconvex, nonsmooth* and *extended-real-valued* functions representing penalties on the inputs, *e.g.*, constraints.

We are interested in simple algorithms for (1), *i.e.*, algorithms that do not involve a doubly iterative procedure, such as SQP methods. One such algorithm is certainly forward-backward splitting (FBS), also known as the proximal gradient method. Let $F : \mathbb{R}^{Nn_u} \to \mathbb{R}^{(N+1)n_x}$ be defined as

$$F(u_0, \dots, u_{N-1}) = (F_0(u), \dots, F_N(u)),$$

where $F_0 \equiv \bar{x}$, while

$$F_{n+1}(u) = f_n(F_n(u), u_n), \quad n = 0, \dots, N-1,$$

and, denoting $u = (u_0, \dots, u_{N-1})$,

$$\begin{aligned} \ell(u) &= \textstyle\sum_{n=0}^{N-1} \ell_n(F_n(u), u_n) + \ell_N(F_N(u)), \\ g(u) &= \textstyle\sum_{n=0}^{N-1} g_n(u_n). \end{aligned}$$

Then, problem (1) can be expressed as

$$\underset{u \in \mathbb{R}^{Nn_u}}{\textbf{minimize}} \ \varphi(u) \equiv \ell(u) + g(u). \tag{2}$$

The FBS scheme is based on simple iterations of the form

$$u^{k+1} \in T_\gamma(u^k) \coloneqq \mathbf{prox}_{\gamma g}\left(u^k - \gamma \nabla \ell(u^k)\right), \tag{3}$$

where $\gamma > 0$ is a stepsize parameter. Here, $\mathbf{prox}_{\gamma g}$ is the (set-valued) proximal mapping of g:

$$\mathbf{prox}_{\gamma g}(u) = \underset{v \in \mathbb{R}^{Nn_u}}{\mathbf{argmin}} \left\{ g(v) + \tfrac{1}{2\gamma}\|v - u\|^2 \right\}.$$

For instance, when g is the indicator of a set the proximal mapping is the Euclidean projection onto the set. We assume that g is simple enough so that the proximal mapping can be evaluated efficiently, and this is true in many examples.

The gradient of ℓ in (3) is efficiently calculated by backward automatic differentiation (also known as reverse mode AD, adjoint method, or backpropagation), see [5].

Iteration (3) is a direct extension of the usual gradient method for problems involving an additional nonsmooth term g. It is widely accepted in the optimization community that the gradient method can be very inefficient: in fact, for nonlinear optimal control problems where $g = 0$ (unconstrained optimal control problems) several more efficient algorithms have been proposed such as nonlinear conjugate gradient or Newton methods, see [2, §2.6]. However, when the additional nonsmooth term g is present, one is left with not many choices other than the proximal gradient method. One option in the case of $g = \delta_C$, where C is a box, is to apply the two-metric projection method of Gafni & Bertsekas [14], the trust-region algorithm of [15], or the limited-memory BFGS algorithm for bound constrained optimization in [16], or more generally, when C is a simple polyhedral set (one that is easy to project onto), the algorithms of [17]. When C has a more complicated structure, extensions of this class of methods become quite complex [18]. When g is a general nonsmooth (perhaps nonconvex) function (such as the sparsity inducing ℓ_1-norm, the sum-of-ℓ_2-norms to induce group sparsity, or the indicator of a nonconvex set such as a finite set of points) then the mentioned algorithms do not apply.

In the present paper we develop an algorithm that requires exactly the same computational oracle as FBS and thus fits embedded applications, but that exploits some curvature information about (1) in order to converge much faster.

A. Handling state constraints

The following more general problem allows to handle cases in which state variables are also subject to constraints:

$$\begin{aligned} \textbf{minimize} \quad & \textstyle\sum_{n=0}^{N-1} \ell_n(x_n, u_n) + g_n(u_n) + h_n(C_n(x_n, u_n)) \\ & + \ell_N(x_N) + h_N(C_N(x_N)) \\ \textbf{subject to} \quad & x_0 = \bar{x} \\ & x_{n+1} = f_n(x_n, u_n), \quad n = 0, \dots, N-1 \end{aligned}$$

where $h_n : \mathbb{R}^{m_n} \to \overline{\mathbb{R}}$, $n = 0, \dots, N$ are proper, closed, convex functions with easily computable proximal mapping and $C_n : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{m_n}$, $n \in 0, \dots, N-1$, and $C_N : \mathbb{R}^{n_x} \to \mathbb{R}^{m_N}$ are smooth mappings. For example, when h_n are indicators of the nonpositive orthant then we are left with a classical state-constrained optimal control problem.

Next, consider $G : \mathbb{R}^{Nn_u} \to \mathbb{R}^{m_0} \times \cdots \times \mathbb{R}^{m_N}$ defined as

$$G(u_0, \dots, u_{N-1}) = (G_0(u), \dots, G_N(u)),$$

where $G_n(u) = C_n(F_n(u), u_n)$ for $n = 0, \dots, N-1$, $G_N(u) = C_N(F_N(u))$, and $h : \mathbb{R}^{m_0} \times \cdots \times \mathbb{R}^{m_N} \to \overline{\mathbb{R}}$ with

$$h(z) = \textstyle\sum_{n=0}^{N-1} h_n(z_n) + h_N(z_n).$$

The problem can now be expressed as

$$\textbf{minimize} \ \ell(u) + g(u) + h(G(u)).$$

Algorithm 1 Backward AD (soft-constrained states)

Inputs: $x_0 \in \mathbb{R}^n$, $u = (u_0, \dots, u_{N-1})$
$\ell(u) \leftarrow 0$
for $n = 0, \dots, N-1$ **do**
  $s_n = \mathbf{prox}_{h_n/\mu}(C_n(x_n, u_n))$
  $q_n = \mu(C_n(x_n, u_n) - s_n)$
  $\ell(u) \leftarrow \ell(u) + \ell_n(x_n, u_n) + h(s_n) + \frac{1}{2\mu}\|q_n\|^2$
  $x_{n+1} = f_n(x_n, u_n)$
$s_N = \mathbf{prox}_{h_N/\mu}(C_N(x_N))$
$q_N = \mu(C_N(x_N) - s_N)$
$\ell(u) \leftarrow \ell(u) + \ell_N(x_N) + h_N(s_N) + \frac{1}{2\mu}\|q_N\|^2$
$p_N = \nabla_{x_N}\ell_N + \nabla_{x_N} C_N q_N$
for $n = N-1, \dots, 0$ **do**
  $p_n = \nabla_{x_n} f_n p_{n+1} + \nabla_{x_n}\ell_n + \nabla_{x_n} C_n q_n$
  $\nabla_{u_n}\ell(u) = \nabla_{u_n} f_n p_{n+1} + \nabla_{u_n}\ell_n + \nabla_{u_n} C_n q_n$

A standard practice in MPC is to include state constraints in the cost function via penalties. The reason for doing so is to avoid ending up with an infeasible optimal control problem which can easily happen in practice due to disturbances and plant-model mismatch. The usual way of doing so is by relaxing state constraints using a quadratic penalty. Taking this approach one step further, we smoothen out h by replacing it with its *Moreau envelope* $h^{1/\mu}$, *i.e.*, the value function of the parametric problem involved in the definition of the proximal mapping. Here μ acts as a penalty parameter: in the case of state constraints of the form $G(u) \in C$, one has $h^{1/\mu}(G(u)) = \frac{\mu}{2}\mathbf{dist}_C^2(G(u))$ and the larger the value of μ, the larger is the penalty for violating the state constraints.

It is well known that the Moreau envelope is smooth when h is proper, closed, convex. In fact its gradient is given by

$$\nabla h^{1/\mu}(z) = \mu(z - \mathbf{prox}_{h/\mu}(z)).$$

Since G is also smooth (as the composition of smooth mappings), the following modified stage costs are smooth

$$\tilde{\ell}_n(u) = \ell_n(F_n(u), u_n) + h_n^{1/\mu}(G_n(u)),\ n = 0, \dots, N-1$$
$$\tilde{\ell}_N(u) = \ell_N(F_N(u)) + h_N^{1/\mu}(G_N(u))$$

and the same holds for the total cost, which we redefine as

$$\ell \leftarrow \textstyle\sum_{n=0}^{N} \tilde{\ell}_n.$$

Therefore soft-state-constrained problems have the same form (2). Alg. 1 can be used to efficiently compute $\nabla\ell$.

Remark II.1. We have considered the case where parameter μ is a scalar for simplicity: Alg. 1 can immediately be adapted, in case h_n are separable, to the case where μ is a vector of parameters, of dimension compatible with the separability structure of h_n. Similarly, the penalty parameter μ can be allowed to depend on n. □

III. Forward-Backward Newton Type Algorithm

First studied for convex problems, FB iterations (3) have been recently shown to converge for problems where both ℓ and g are nonconvex [7]: if ℓ has L_ℓ-Lipschitz continuous gradient and g is lower-bounded, then for any $\gamma \in (0, 1/L_\ell)$ all accumulation points $u^\star$ of sequences complying with (3) are γ-*critical*, *i.e.*, they satisfy the condition

$$u^\star \in \mathbf{prox}_{\gamma g}(u^\star - \gamma\nabla\ell(u^\star)).$$

Moreover, if $\ell + g$ is a Kurdyka-Łojasiewicz function — a mild property satisfied by all subanalytic functions, for instance — then any bounded sequence (3) is globally convergent to a unique critical point.

Because of such favorable properties, and the fact that in many problems the proximal mapping is available in close form, FBS has been extensively employed and studied. The downside of such simple algorithm is its slow tail convergence, being it Q-linear at best and with Q-factor typically close to one when the problem is ill-conditioned. The employment of variable metrics, *e.g.*, coming from Newton-type schemes, can dramatically improve and robustify the convergence, at the cost of prohibitively complicating the proximal steps, which would require inner procedures possibly as hard as solving the original problem itself.

A. Newton-type methods on generalized equations

Instead of directly addressing the minimization problem, one could target the complementary problem of finding critical points by solving the inclusion (*generalized equation*)

$$\text{find } u^\star \text{ such that } 0 \in R_\gamma(u^\star) := \tfrac{1}{\gamma}[u - T_\gamma(u)], \quad (4)$$

Here R_γ is the (set-valued) *fixed-point residual*. Under very mild assumptions, R_γ is well-behaved close to critical points, and when close to a solution problem (4) reduces to a classical equation, as opposed to generalized equation. This motivates addressing the problem using Newton-type methods

$$u^{k+1} = u^k - H_k R_\gamma(u^k), \quad (5)$$

where H_k are invertible operators that, ideally, capture curvature information of R_γ and enable superlinear or quadratic convergence when close enough to a solution. In quasi-Newton schemes, H_k is a linear operator recursively updated so as to satisfy the (inverse) secant condition

$$u^{k+1} - u^k = H_{k+1}\big(R_\gamma(u^{k+1}) - R_\gamma(u^k)\big),$$

and under mild differentiability assumptions at a candidate limit point $u^\star$, local superlinear convergence is achieved provided that the Dennis-Moré condition

$$\lim_{k\to\infty} \frac{\|R_\gamma(u^k) - JR_\gamma(u^\star)d^k\|}{\|d^k\|} = 0 \quad (6)$$

is satisfied, where $d^k = -H_k R_\gamma(u^k)$.

B. Forward-backward envelope

The drawback of iterations of the type (5) is that convergence can only be guaranteed provided that u^0 is close enough to a solution. In fact, without globalization strategies such type of methods are well known to even possibly diverge. In [10] a globalization technique is proposed, based on the *forward-backward envelope* (FBE) [8] (initially derived for convex problems [19], [9]). The FBE is an exact, continuous, real-valued penalty function for (2), defined as

$$\varphi_\gamma(u) := \ell(u) - \tfrac{\gamma}{2}\|\nabla\ell(u)\|^2 + g^\gamma(u - \gamma\nabla\ell(u)). \quad (7)$$

Proposition III.1 ([10]). *For any $\gamma > 0$, φ_γ is a strictly continuous function satisfying*

(i) $\varphi_\gamma \leq \varphi$;

(ii) $\varphi(\bar{u}) \leq \varphi_\gamma(u) - \frac{1-\gamma L_\ell}{2\gamma}\|u - \bar{u}\|^2$ *for any* $\bar{u} \in T_\gamma(u)$.

In particular,

(iii) $\varphi(u) = \varphi_\gamma(u)$ *for any* $u \in \mathbf{fix}\, T_\gamma$;

(iv) $\inf \varphi = \inf \varphi_\gamma$ *and* $\operatorname{argmin} \varphi = \operatorname{argmin} \varphi_\gamma$ *for any* $\gamma < 1/L_\ell$.

Proof. See [20]. □

By strict continuity, via Rademacher's theorem [21, Thm. 9.60], $\nabla\ell$ and φ_γ are almost everywhere differentiable with

$$\nabla\varphi_\gamma(u) = Q_\gamma(u)R_\gamma(u), \quad \text{where} \quad Q_\gamma(u) \coloneqq I - \gamma\nabla^2\ell(u),$$

see [10]. Matrices $Q_\gamma(u)$ are symmetric and defined for almost any u; if $\gamma < 1/L_\ell$, then $Q_\gamma(u)$ is also positive definite wherever it exists. If ℓ is twice differentiable at a critical point $u^\star$ and $\mathbf{prox}_{\gamma g}$ is differentiable at $u^\star - \gamma\nabla\ell(u^\star)$, then φ_γ is twice differentiable at $u^\star$ with Hessian [10]

$$\nabla^2\varphi_\gamma(u^\star) = Q_\gamma(u^\star)JR_\gamma(u^\star). \tag{8}$$

A sufficient condition for $\mathbf{prox}_{\gamma g}$ to comply with this requirement involves a mild property of prox-regularity and twice epi-differentiability, see [21, §13].

Theorem III.2 (Strong local minimality. [10]). *Let $\gamma < 1/L_\ell$ and suppose that $\nabla\ell$ and $\mathbf{prox}_{\gamma g}$ are differentiable at a critical point $u^\star$ and at $u^\star - \gamma\nabla\ell(u^\star)$, respectively. Then, $u^\star$ is a strong local minimum for φ iff it is a strong local minimum for φ_γ, in which case $\nabla^2\varphi_\gamma(u^\star)$ is positive definite and $JR_\gamma(u^\star)$ is invertible.*

C. A superlinearly convergent algorithm based on FBS steps

To the best of our knowledge, [10] proposes the first algorithm with superlinear convergence guarantees that is entirely based on forward-backward iterations. In this work, we propose **PANOC** (Proximal Averaged Newton-type method for Optimal Control), a new linesearch method for problem (2), which is even simpler than the one of [10], yet it maintains all the favorable convergence properties. After a quick glance at the favorable properties of the FBE and its kinship with FBS, the methodology of the proposed scheme is elementary. At each iteration, a forward-backward element $\bar{u}^k$ is computed. Then, a step is taken along a convex combination of the "nominal" FBS update direction $-\gamma r^k$ and a candidate fast direction d^k. By appropriately *averaging* between the two directions we can ensure sufficient decrease of the FBE, enabling global convergence. When close to a solution, fast directions will take over and the iterations reduce to $u^{k+1} = u^k + d^k$. The next results rigorously show these claims.

Theorem III.3 (Global subsequential convergence). *Consider the iterates generated by* **PANOC**. *Then, $r^k \to 0$ square-summably, and the sequences $(u^k)_{k\in\mathbb{N}}$ and $(\bar{u}^k)_{k\in\mathbb{N}}$ have the same cluster points, all satisfying the necessary condition for local minimality $u \in \mathbf{prox}_{\gamma g}(u - \gamma\nabla\ell(u))$.*

Proof. See [20]. □

Algorithm PANOC.

Inputs: $\gamma \in (0, 1/L_\ell)$, $\sigma \in (0, \gamma\frac{1-\gamma L_\ell}{2})$, $u_0 \in \mathbb{R}^{Nn_u}$.

1: **for** $k = 0, 1, \ldots$ **do**
2: Compute $\nabla\ell(u^k)$ using Alg. 1
3: $\bar{u}^k = \mathbf{prox}_{\gamma g}\left(u^k - \gamma\nabla_u\ell(u^k)\right)$, $r^k = \frac{u^k - \bar{u}^k}{\gamma}$
4: Let $d^k = -H_k r^k$ for some matrix $H_k \in \mathbb{R}^{Nn_u \times Nn_u}$
5: $u^{k+1} = u^k - (1-\tau_k)\gamma r^k + \tau_k d^k$, where τ_k is the largest in $\{(1/2)^i \mid i \in \mathbb{N}\}$ such that

$$\varphi_\gamma(u^{k+1}) \leq \varphi_\gamma(u^k) - \sigma\|r^k\|^2 \tag{9}$$

Remark III.4 (Lipschitz constant L_ℓ). In practice, no prior knowledge of the Lipschitz constant L_ℓ is required for **PANOC**. In fact, replacing L_ℓ with an initial estimate $L > 0$, the following instruction can be added right after step 3:

3bis: **if** $\ell(\bar{u}^k) > \ell(u^k) - \gamma\langle\nabla\ell(u^k), r^k\rangle + \frac{L}{2}\|\gamma r^k\|^2$ **then** $\gamma \leftarrow \gamma/2$, $L \leftarrow 2L$, $\sigma \leftarrow \sigma/2$ and go to step 3.

The above condition will fail to hold as soon as $L \geq L_\ell$ [2, Prop. A.24], and consequently L is incremented only a finite number of times. Therefore, there exists an iteration k_0 starting from which γ and σ are constant, and all the results of the paper remain valid if such a strategy is implemented.

Moreover, since $\bar{u}^k \in \mathbf{dom}\, g$ by construction, if g has bounded domain and the selected directions d^k are bounded (as it is the case for any "reasonable" implementation), it suffices that $\nabla\ell$ is *locally* Lipschitz-continuous (*i.e.*, strictly continuous), and as such any $\ell \in C^2$ would fit the requirement. In fact, in such case all the sequences $(u^k)_{k\in\mathbb{N}}$ and $(\bar{u}^k)_{k\in\mathbb{N}}$ are contained in a compact enlargement Ω of $\mathbf{dom}\,\partial g$, and L_ℓ can be then taken as $\mathbf{lip}_\Omega(\nabla\ell)$, or adaptively retrieved in practice as indicated above. This is the typical circumstance in (N)MPC where g encodes input constraints, which in realistic applications are bounded. □

Each evaluation of φ_γ in the left-hand side of the linesearch condition (9) requires one forward-backward step; $\varphi_\gamma(u^k)$ on the right-hand side, instead, is available from the previous iteration. In particular, in the best case of stepsize $\tau_k = 1$ being accepted, each iteration requires exactly one forward-backward step. Under mild assumptions, this is the case when directions d^k satisfy the Dennis-Moré condition (6), as shown in the following result. This shows that the FBE does not prevent superlinear convergence of **PANOC** when Newton-type directions are used: eventually, unit stepsize is accepted and **PANOC** reduces to (5). This is in stark contrast with the well known drawback of classical nonsmooth exact penalties (the so-called Maratos effect [2, §5.3]).

Theorem III.5 (Superlinear convergence). *Suppose that in* **PANOC** *$u^k \to u^\star$, for a strong local minimum $u^\star$ of φ at which R_γ and $\nabla\varphi_\gamma$ are strictly differentiable. If $(H_k)_{k\in\mathbb{N}}$ satisfies the Dennis-Moré condition (6), then $\tau_k = 1$ is eventually always accepted and $u^k \to u^\star$ at superlinear rate.*

Proof. See [20]. □

Strict differentiability of R_γ and $\nabla\varphi$ does not require any

smoothness condition on the nonsmooth function g. In fact, the required conditions hold as long as g is *prox-regular* and has a generalized quadratic *epigraphical Hessian* at the limit point; prox-regularity is a mild property enjoyed, for instance, by any convex function or a function whose effective domain is a discrete set, and similarly functions complying with the required generalized second-order properties are ubiquitous in optimization, see [21], [10] and references therein. For example, *partly smooth* functions are a comprehensive class of functions for which such properties hold; in fact, if the critical point $u^\star$ satisfies the qualification $-\nabla\ell(u^\star) \in \mathbf{relint}\,\partial g(u^\star)$ and g is prox-regular at $u^\star$, then $\mathbf{prox}_{\gamma g}$ is differentiable around $u^\star - \gamma\nabla\ell(u^\star)$, see [22].

The Dennis-Moré condition is enjoyed (under differentiability assumptions at the limit point) by directions generated with quasi-Newton schemes, the BFGS method being a prominent example. Because of the problems size, in §IV we will show the efficiency of **PANOC** with its limited-memory variant L-BFGS: this does not require storing the matrices H_k, but instead keeps memory of a small number of pairs $s_k = u^{k+1} - u^k$ and $y^k = r^{k+1} - r^k$, and retrieves $d = -H_k r^k$ by simply performing scalar products.

IV. Numerical Simulations

To test the efficacy of the proposed algorithm we consider a system composed of a sequence of masses connected by springs [23], [24]. The chain is composed by M masses: one end is connected to the origin, while a handle on the other end allows to control the chain. Let us denote by $p^i(t) \in \mathbb{R}^3$ the position of the i-th mass at time t, for $i = 1, \dots, M+1$, where $p^{M+1}(t)$ is the position of the control handle. The control action at each time instant is denoted as $u(t) = \dot{p}^{M+1}(t) \in \mathbb{R}^3$, *i.e.*, we control the velocity of the handle. Each body in the chain has mass m, and the springs have constant D and rest length L. By Hook's law we obtain the dynamics [23]:

$$\ddot{p}^i = \tfrac{1}{m}(F_{i,i+1} - F_{i-1,i}) + a,$$
$$F_{i,i+1} = D\left(1 - \tfrac{L}{\|p^{i+1} - p^i\|}\right)(p^{i+1} - p^i).$$

where $a = (0, 0, -9.81)$ is the acceleration due to gravity. Denoting v^i the velocity of mass i, the state vector is

$$x(t) = (p^1(t), \dots, p^{M+1}(t), v^1(t), \dots, v^M(t)).$$

In the three-dimensional space there are $n_x = 3(2M+1)$ state variables and $n_u = 3$ input variables, and

$$\dot{x} = f_c(x, u) = (v^1, \dots, v^M, u, \ddot{p}^1, \dots, \ddot{p}^M).$$

A. Simulation scenario

An equilibrium state of the system was computed with the control handle positioned at a given $p_{\text{end}} \in \mathbb{R}^3$. This was perturbed by applying a constant input $u = (-1, 1, 1)$ for 1 second, to obtain the starting position of the chain. The goal is to drive the system back to the reference equilibrium state: this is achieved by solving, for $T > 0$

$$\begin{aligned} \underset{x,u}{\mathbf{minimize}} \quad & L_c(T) = \textstyle\int_0^T \ell_c(x(t), u(t))dt \\ \mathbf{subject\ to} \quad & \dot{x} = f_c(x, u) \end{aligned} \tag{10}$$

where

$$\ell_c(x, u) = \beta\|p^{M+1} - p_{\text{end}}\|_2^2 + \gamma\textstyle\sum_{i=1}^M \|v^i\|_2^2 + \delta\|u\|_2^2. \tag{11}$$

To discretize (10) we consider a sampling time t_s such that $T = Nt_s$ and piecewise constant input u accordingly: for $n = 0, \dots, N-1$, $u(t) = u_n$ for all $t \in [nt_s, (n+1)t_s)$. Then $L_c(T) = \sum_{n=0}^{N-1} \int_{nt_s}^{(n+1)t_s} \ell_c(x(t), u_n)dt$: the problem is cast into the form (1) by discretizing the integrals in the sum, and the system dynamics, by setting

$$\ell_n(x_n, u_n) \approx \textstyle\int_{nt_s}^{(n+1)t_s} \ell_c(x(t), u_n)dt, \tag{12a}$$
$$f_n(x_n, u_n) \approx \textstyle\int_{nt_s}^{(n+1)t_s} f_c(x(t), u_n)dt, \tag{12b}$$

with the initial condition $x(nt_s) = x_n$, $n = 0, \dots, N-1$. Furthermore, we constrain the states and inputs by setting g_n and h_n as the indicator functions of the feasible sets as

$$g_n(u) = \delta_{\|\cdot\|_\infty \le 1}(u),$$
$$h_n(C_n(x, u)) = \textstyle\sum_{i=1}^{M+1} \delta_{\ge -0.1}(x_2^i).$$

Since h_n is separable with respect to the different masses, we smoothen it by associating a parameter μ_i to each component (see §II-A and Rem. II.1 in particular):

$$h_n^{1/\mu}(C_n(x, u)) = \textstyle\sum_{i=1}^{M+1} \frac{\mu_i}{2}\left(\mathbf{min}\left\{0, p_2^i + 0.1\right\}\right)^2. \tag{13}$$

In the simulations we have used $T = 4$ seconds and a sampling time $t_s = 0.1$ seconds, which gives a prediction horizon $N = 40$. Integrals (12) were approximated with a one-step 4th-order Runge-Kutta method. We used CasADi [11] to implement the dynamics and cost function, and to efficiently evaluate their Jacobian and gradient. The model parameters were set as $M = 5$, $m = 0.03$ (kg), $D = 0.1$ (N/m), and $L = 0.033$ (m). In (11) we set $\beta = 1$, $\gamma = 1$, and $\delta = 0.01$. The coefficients for the soft state constraints (13) were set as $\mu_1 = \mu_2 = \mu_3 = 10^2$, $\mu_4 = \mu_5 = \mu_6 = 10$.

B. Results

We simulated the system for 15 seconds using different solvers. In **PANOC** we computed d^k in step 4 using the L-BFGS method with memory 10 (see discussion in §III-C). Furthermore, we applied FBS, MATLAB's FMINCON (using an SQP algorithm), IPOPT (interior-point method) to both the single- and multiple-shooting formulations, and to the problem with hard state constraints. We did not apply FMINCON to the multiple-shooting problem, as doing so performed considerably worse. Fig. 1 shows the convergence of the fixed-point residual $\|r^k\|_\infty$ for FBS and **PANOC** for the first problem of the sequence: there we have solved the problem to medium/high accuracy for comparison purposes. In practice, we have noticed that good closed loop performance is obtained with more moderate accuracy: we ran closed-loop simulations terminating **PANOC** and FBS as soon as $\|r^k\|_\infty \le 10^{-3}$. The other solvers were run with default options. The CPU times during the simulation are shown in Fig. 1. **PANOC** outperforms the other considered methods in this example, and greatly accelerates over FBS: this is particularly evident early in the simulation, when the system is far from equilibrium. The effect of soft state constraints on the dynamics is shown in Fig. 2, where the trajectory of

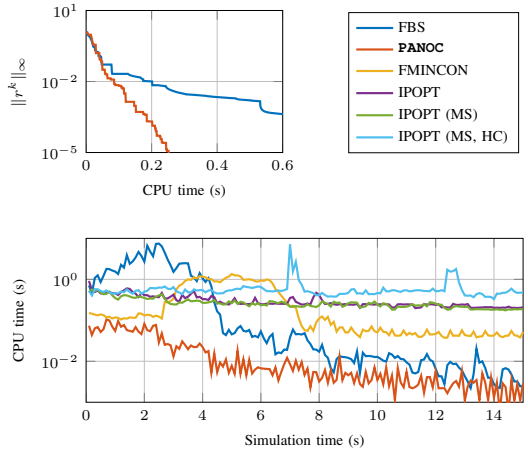

Fig. 1: *(Top) Convergence of FBS and* **PANOC** *in the first problem of the closed-loop simulation: the algorithms were executed here to medium/high accuracy for comparison purposes. (Bottom) CPU times of the solvers in the closed-loop simulation ("MS": multiple shooting, "HC": hard constraints).*

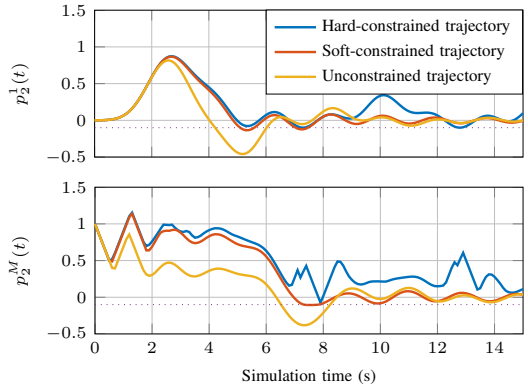

Fig. 2: *Effect of the soft state constraint terms on the trajectory of the masses* 1 *and* M *in the closed loop simulation.*

two masses during the simulation is compared to the hard-constrained and unconstrained cases. Apparently, using soft state constraints improves considerably the solution time of the problem, without sacrificing closed loop performance.

V. Conclusions

This paper presents **PANOC**, a new algorithm for solving nonlinear constrained optimal control problems typically arising in MPC. The algorithm is simple, exploits problem structure, does not require solution of a quadratic program at every iteration and yet can be shown to be superlinearly convergent under mild assumptions. Using L-BFGS directions in the algorithm was shown to perform favorably against state-of-the-art NLP solvers in a benchmark example.

There are several topics for future research: *(i)* semismooth Newton directions [19] that fully exploit the problem structure enabling quadratic convergence rates, *(ii)* more rigorous handling of state constraints by embedding the algorithm in a proximal augmented Lagrangian framework, *(iii)* a real-time iteration scheme where the algorithm is warm-started by exploiting sensitivity information for the fixed point residual and *(iv)* a code generation tool for embedded applications.

References

[1] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2006.

[2] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2016.

[3] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast motions in biomechanics and robotics*. Springer, 2006, pp. 65–93.

[4] R. Quirynen, "Numerical simulation methods for embedded optimization," Ph.D. dissertation, KU Leuven, 2017.

[5] J. C. Dunn and D. P. Bertsekas, "Efficient dynamic programming implementations of Newton's method for unconstrained optimal control problems," *Journal of Optimization Theory and Applications*, vol. 63, no. 1, pp. 23–38, 1989.

[6] S. J. Wright, "Solution of discrete-time optimal control problems on parallel computers," *Parallel Computing*, vol. 16, no. 2-3, pp. 221–237, 1990.

[7] H. Attouch, J. Bolte, and B. F. Svaiter, "Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods," *Mathematical Programming*, vol. 137, no. 1, pp. 91–129, 2013.

[8] P. Patrinos and A. Bemporad, "Proximal Newton methods for convex composite optimization," in *IEEE Conference on Decision and Control*, 2013, pp. 2358–2363.

[9] L. Stella, A. Themelis, and P. Patrinos, "Forward-backward quasi-Newton methods for nonsmooth optimization problems," *Computational Optimization and Applications*, vol. 67, no. 3, pp. 443–487, 2017.

[10] A. Themelis, L. Stella, and P. Patrinos, "Forward-backward envelope for the sum of two nonconvex functions: Further properties and nonmonotone line-search algorithms," *arXiv:1606.06256*, 2016.

[11] J. Andersson, J. Åkesson, and M. Diehl, "CasADi: A symbolic package for automatic differentiation and optimal control," in *Recent advances in algorithmic differentiation*. Springer, 2012, pp. 297–307.

[12] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[13] A. Themelis, L. Stella, and P. Patrinos, "Douglas-Rachford splitting and ADMM for nonconvex optimization: new convergence results and accelerated versions," *arXiv:1709.05747*, 2017.

[14] E. M. Gafni and D. P. Bertsekas, "Two-metric projection methods for constrained optimization," *SIAM Journal on Control and Optimization*, vol. 22, no. 6, pp. 936–964, 1984.

[15] C.-J. Lin and J. J. Moré, "Newton's method for large bound-constrained optimization problems," *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 1100–1127, 1999.

[16] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[17] P. H. Calamai and J. J. Moré, "Projected gradient methods for linearly constrained problems," *Mathematical Programming*, vol. 39, no. 1, pp. 93–116, 1987.

[18] J. C. Dunn, "A projected Newton method for minimization problems with nonlinear inequality constraints," *Numerische Mathematik*, vol. 53, no. 4, pp. 377–409, 1988.

[19] P. Patrinos, L. Stella, and A. Bemporad, "Forward-backward truncated Newton methods for large-scale convex composite optimization," *arXiv:1402.6655*, 2014.

[20] L. Stella, A. Themelis, P. Sopasakis, and P. Patrinos, "A simple and efficient algorithm for nonlinear model predictive control," *arXiv:1709.06487*, 2017.

[21] R. T. Rockafellar and R. J. Wets, *Variational analysis*. Springer, 2011, vol. 317.

[22] A. S. Lewis, "Active sets, nonsmoothness, and sensitivity," *SIAM Journal on Optimization*, vol. 13, no. 3, pp. 702–725, 2002.

[23] L. Wirsching, H. G. Bock, and M. Diehl, "Fast NMPC of a chain of masses connected by springs," in *IEEE International Conference on Control Applications*, 2006, pp. 591–596.

[24] M. Vukov, A. Domahidi, H. J. Ferreau, M. Morari, and M. Diehl, "Auto-generated algorithms for nonlinear model predictive control on long and on short horizons," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. IEEE, 2013, pp. 5113–5118.