

# Exact Complexity Certification of a Nonnegative Least-Squares Method for Quadratic Programming

Daniel Arnström<sup>1</sup>, Alberto Bemporad<sup>2</sup>, *Fellow, IEEE*, and Daniel Axehill<sup>1</sup>

**Abstract**—In this letter we propose a method to exactly certify the complexity of an active-set method which is based on reformulating strictly convex quadratic programs to nonnegative least-squares problems. The exact complexity of the method is determined by proving the correspondence between the method and a standard primal active-set method for quadratic programming applied to the dual of the quadratic program to be solved. Once this correspondence has been established, a complexity certification method which has already been established for the primal active-set method is used to also certify the complexity of the nonnegative least-squares method. The usefulness of the proposed method is illustrated on a multi-parametric quadratic program originating from model predictive control of an inverted pendulum.

**Index Terms**—Optimization algorithms, predictive control for linear systems.

## I. INTRODUCTION

AN OPTIMIZATION problem has to be solved in each time-instant when model predictive control (MPC) is used for control. The optimization problems in question are often quadratic programs (QPs) and to be able to use MPC in embedded systems, the employed QP solvers need to be simple, fast and have real-time guarantees. Popular methods for solving QPs originating from MPC are interior-point methods [1], [2], gradient projection methods [3], [4], [5], the alternating method of multipliers (ADMM) [6] and active-set methods [7], [8], [9], [10], [11]. The active-set method in [11], which is based on reformulating strict convex quadratic programs to nonnegative least squares (NNLS) problems, is simple to code and has been shown to be efficient for solving

QPs originating from MPC. However, since it is an active-set method, its complexity can be exponential in the worst case.

To be able to provide tight real-time guarantees for active-set methods, complexity certification methods which determine the worst-case behaviour for the active-set methods in [7], [8] and [10] have been presented in [12], [13] and [14], respectively. For a given MPC problem, these methods determine *exactly* which subproblems, i.e., systems of linear equations, that have to be solved to find the solution, for every possible QP that needs to be solved. A unifying complexity certification framework for a class of standard active-set methods, which covers both the methods from [12] and [13], is available in [15].

In this letter we extend the possibility to also certify the complexity of the efficient QP method presented in [11], adding to its simplicity and efficiency the possibility to determine its exact complexity, increasing its practical applicability. This is done by proving that the working-set changes of the QP method are equivalent to a standard primal active-set method applied to the dual of the QP to be solved. This equivalence allows direct use of the complexity certification framework in [15]. The main focus of this letter is, hence, not to devise another complexity certification method from scratch, but to relate the method presented in [11] to the active-set method considered in [15], for which there exists a complexity certification method. In summary, the main contribution of this letter is proving the equivalence between the QP methods in [11] and [15], and from this equivalence the technical contribution of a method which certifies the exact complexity of the QP method in [11] follows.

## II. PROBLEM FORMULATION

Consider a multi-parametric quadratic program (mpQP) in the form

$$\begin{aligned} & \underset{x}{\text{minimize}} && \frac{1}{2}x^T Hx + (f^T + \theta^T f_\theta^T)x \\ & \text{subject to} && Ax \leq b + W\theta, \end{aligned} \quad (1)$$

where  $x \in \mathbb{R}^n$  and the parameter  $\theta \in \Theta_0 \subseteq \mathbb{R}^p$ , with  $\Theta_0$  being a polytope (such as a box). The mpQP is given by  $A \in \mathbb{R}^{m \times n}$ ,  $b \in \mathbb{R}^m$ ,  $W \in \mathbb{R}^{m \times p}$ ,  $f \in \mathbb{R}^n$ ,  $f_\theta \in \mathbb{R}^{n \times p}$ , and  $H \in \mathbb{S}_{++}^n$ . The minimizer of (1), given  $\theta$ , is denoted  $x^*(\theta)$ . It is well-known

Manuscript received March 17, 2020; revised April 30, 2020; accepted May 17, 2020. Date of publication June 1, 2020; date of current version June 12, 2020. This work was supported by the Swedish Research Council (VR) under Contract 2017-04710. Recommended by Senior Editor F. Dabbene. (Corresponding author: Daniel Arnström.)

Daniel Arnström and Daniel Axehill are with the Division of Automatic Control, Linköping University, 58183 Linköping, Sweden (e-mail: daniel.arnstrom@liu.se; daniel.axehill@liu.se).

Alberto Bemporad is with the Department of Computer Science and Engineering, IMT School for Advanced Studies Lucca, 55100 Lucca, Italy (e-mail: alberto.bemporad@imtlucca.it).

Digital Object Identifier 10.1109/LCSYS.2020.2998953

that a linear MPC problem can be cast in the form (1), where the parameter  $\theta$  contains the measured/estimated states and reference signals [16].

In [11] it is shown that by introducing

$$M \triangleq A\mathcal{L}^{-1}, \quad d(\theta) \triangleq b + W\theta + AH^{-1}(f + f_\theta\theta), \quad (2)$$

where  $\mathcal{L}$  is the Cholesky factor of  $H$ , (1) can be restated as the nonnegative least-squares problem

$$\underset{y \geq 0}{\text{minimize}} \quad \frac{1}{2} \left\| \begin{bmatrix} M^T \\ d(\theta)^T \end{bmatrix} y + \begin{bmatrix} 0 \\ \gamma \end{bmatrix} \right\|_2^2 \quad (3)$$

where  $\gamma$  is any positive scalar,  $M \in \mathbb{R}^{m \times n}$ ,  $d(\theta) \in \mathbb{R}^m$  and  $y \in \mathbb{R}^m$ . Furthermore, the relationship between  $x^*(\theta)$  and the minimizer of (3),  $y^*(\theta)$ , is

$$x^*(\theta) = -H^{-1} \left( f + f_\theta\theta + \frac{1}{\gamma + d(\theta)^T y^*(\theta)} A^T y^*(\theta) \right) \quad (4)$$

*Remark 1:* It is straightforward to extend the ideas in this letter to also handle equality constraints in (1) by using the results in [17]. However, for the sake of a clean presentation, we will only consider inequality constraints.

### A. Notation

Since the algorithm to be studied is iterative, we use a subscript  $k$  to denote the value at iteration  $k$  for quantities that change between iterations, e.g.,  $y_k$  denotes the value of  $y$  at iteration  $k$ . Of importance is also the so-called working set  $\mathcal{W}_k$  which contains a subset of the components of  $y_k$  that are free to vary. Likewise, the set of components that are not in  $\mathcal{W}_k$  is denoted  $\bar{\mathcal{W}}_k$  and contains components that are fixed at zero. For indexing of matrices,  $[N]_i$  denotes the  $i$ th row of matrix  $N$  and  $[N]_{\mathcal{W}_k}$  denotes the submatrix obtained by extracting the rows of  $N$  indexed by  $\mathcal{W}_k$ . The shorter notation  $N_k \triangleq [N]_{\mathcal{W}_k}$  for matrices that do not change between iterations is also introduced for convenience.

### B. Nonnegative Least-Squares Method

The nonnegative least-squares method for solving (1) presented in [11] is described briefly below and summarized in Algorithm 1. For a more detailed description, see [11] or [18, Sec. 23.3]. The main objective of Algorithm 1 is to retain nonnegativity of the iterate  $y$  while updating the working set  $\mathcal{W}$ . At iteration  $k$ , the least-squares (LS) problem

$$\begin{aligned} & \underset{y}{\text{minimize}} \quad \frac{1}{2} \left\| \begin{bmatrix} M^T \\ d(\theta)^T \end{bmatrix} y + \begin{bmatrix} 0 \\ \gamma \end{bmatrix} \right\|_2^2 \\ & \text{subject to} \quad [y]_i = 0, \quad i \in \bar{\mathcal{W}}_k \end{aligned} \quad (5)$$

defined by the current working set  $\mathcal{W}_k$  is solved, with the solution of (5) being denoted  $y_k^*$ . The iterate  $y_k$  is then updated to  $y_{k+1}$  by a line search from  $y_k$  to  $y_k^*$ . To retain nonnegativity, the first component of  $y_k$  which becomes negative during this line search is removed from  $\mathcal{W}_k$ . If no such component exists, i.e., if  $y_k^* \geq 0$ , global optimality is checked for  $y_k^*$  by investigating the dual variable  $w_k$ . If  $w_k$  is nonnegative, a global optimum has been found, otherwise, the index of the most negative component of  $w_k$  is added to  $\mathcal{W}$ . When the working

### Algorithm 1 Given $\theta$ , Solve the mpQP (1) With NNLS [11]

```

1:  $v \leftarrow \mathcal{L}^{-T}(f_\theta\theta + f)$ ;  $d \leftarrow (b + W\theta) + Mv$ 
2:  $k \leftarrow 1$ ;  $\mathcal{W}_k \leftarrow \emptyset$ ;  $y_k \leftarrow 0$ 
3: while true do
4:    $y_k^* \leftarrow$  solution to least squares problem (5)
5:   if  $y_k^* \geq 0$  then
6:      $w_k \leftarrow M(M_k^T[y_k^*]_{\mathcal{W}_k}) + (\gamma + d_k^T[y_k^*]_{\mathcal{W}_k})d$ 
7:     if  $w_k \geq -(\gamma + d_k^T[y_k^*]_{\mathcal{W}_k})\epsilon$  then
8:       go to step 15
9:     else  $i \leftarrow \arg \min_{i \in \bar{\mathcal{W}}_k} [w_k]_i$ ,  $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \cup \{i\}$ 
10:     $y_{k+1} \leftarrow y_k^*$ 
11:   else  $l \leftarrow \arg \min_{h \in \mathcal{W} : [y_k^*]_h < 0} \left\{ \frac{[y_k]_h}{[y_k]_h - [y_k^*]_h} \right\}$ 
12:     $\alpha_k \leftarrow \frac{[y]_l}{[y]_l - [y_k^*]_l}$ ;  $y_{k+1} \leftarrow y_k + \alpha_k(y_k^* - y_k)$ 
13:     $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{l\}$ 
14:    $k \leftarrow k + 1$ 
15: return  $\lambda^* \leftarrow \frac{y_k^*}{\gamma + d_k^T y_k^*}$ ,  $x^* \leftarrow -\mathcal{L}^{-1}(M_k^T \lambda_k^* + v)$ ,  $\mathcal{W}_k$ 

```

set has been updated, another LS problem defined by the new working set is solved and the steps above are repeated until global optimality is reached.

The choice of  $\gamma$  is not critical since any  $\gamma > 0$  is sufficient for the algorithm to work. In [18, Sec. 23]  $\gamma = 1$  is used, and in [11]  $\gamma$  is adaptively updated by adding or removing the absolute value of  $[d]_i$  when  $i$  is added or removed from  $\mathcal{W}$ , respectively. In this letter we consider  $\gamma$  to be constant for simplicity. However, the results also extends to an adaptively changing  $\gamma$  since the working-set changes produced by Algorithm 1 are independent of  $\gamma$ .

*Remark 2:* The presentation of Algorithm 1 is slightly modified compared with [11] to make the definition of an iteration in the algorithm clearer. Furthermore, some checks for infeasibility that are included in [11] have been omitted to clean up the algorithm, i.e., we assume that (1) is primal feasible for all  $\theta$  of interest. This condition can be immediately verified, for example, by checking that QP (1) is feasible for all vertices  $\theta_i$  of  $\Theta_0$ , as is shown in Lemma 1.

*Lemma 1:* Let  $\{\theta_i\}_{i=1}^M$  be the vertices of the polytope  $\Theta_0$  and let  $X_i \triangleq \{x \in \mathbb{R}^n : Ax \leq b + W\theta_i\}$  be the feasible set for (1) when  $\theta = \theta_i$ . Then problem (1) is feasible  $\forall \theta \in \Theta_0 \Leftrightarrow X_i \neq \emptyset, \forall i \in \{1, \dots, M\}$ .

*Proof:* Since  $\theta_i \in \Theta_0, \forall i$ , the left-to-right implication follows immediately. For the right-to-left implication we have, since  $\Theta_0$  is convex,  $\theta = \sum_{i=1}^M \alpha_i \theta_i, \sum \alpha_i = 1, \alpha_i \geq 0$ . Let  $x_i \in X_i$  and consider  $x = \sum_{i=1}^M \alpha_i x_i$ . Then  $Ax = \sum \alpha_i Ax_i \leq \sum \alpha_i (b + W\theta_i) = b + W \sum \alpha_i \theta_i = b + W\theta$ . ■

## III. PROPERTIES OF NNLS ALGORITHM

This section describes properties of Algorithm 1 that will be central in Section IV, where a complexity certification method for Algorithm 1 is outlined.

Analyzing the parametric behaviour for Algorithm 1 does not immediately follow from previous work on the topic since there is parameter dependence in the quadratic term in the

objective function of (3) due to  $d(\theta)$ . All of the complexity certification methods in [13], [14] and [15] rely on parameter dependence only appearing in linear terms. Hence, the main objective in this letter will be to disentangle the parameter dependence in the quadratic term and to show that Algorithm 1 will be equivalent, in terms of working-set changes, to another algorithm which operates on a problem with only parameter dependence in the linear term. This disentanglement is done by considering  $\lambda$  which is a linear fractional transformation of  $y$  defined as

$$\lambda \triangleq \frac{y}{\gamma + d^T y}. \quad (6)$$

The following scalars will also prove useful

$$\sigma_k \triangleq \gamma + d^T y_k^*, \quad \rho_k \triangleq \gamma + d^T y_k. \quad (7)$$

With these scalars we have  $y_k = \rho_k \lambda_k$  and  $y_k^* = \sigma_k \lambda_k^*$  if  $\rho_k \neq 0$  and  $\sigma_k \neq 0$ , respectively.

For clarity, we initially deduce properties of Algorithm 1 under the assumption that  $M_k M_k^T$  is nonsingular. In Section III-D we discuss properties of the algorithm in the singular case.

### A. Least-Squares Subproblems

The solution  $y_k^*$  to the subproblem (5) can be found by solving the following KKT-system

$$\begin{pmatrix} MM^T + dd^T & I_{\mathcal{W}_k}^T \\ \bar{\mathcal{W}}_k & 0 \end{pmatrix} \begin{pmatrix} y_k^* \\ w_k \end{pmatrix} = \begin{pmatrix} -\gamma d \\ 0 \end{pmatrix}, \quad (8)$$

where  $I$  is the  $m \times m$  identity matrix. Equation (8) has the solution

$$y_k^* = -\gamma I_k^T (M_k M_k^T + d_k d_k^T)^{-1} d_k. \quad (9)$$

Recall from Section II-A that  $M_k$ ,  $d_k$  and  $I_k$  is shorthand notation for submatrices obtained when indexing with  $\mathcal{W}_k$ , i.e.,  $[M]_{\mathcal{W}_k}$ ,  $[d]_{\mathcal{W}_k}$  and  $[I]_{\mathcal{W}_k}$ , respectively. Another way of finding the solution to (5) is to directly set all components of  $y_k^*$  that are not in  $\mathcal{W}_k$  to zero, resulting in an unconstrained optimization problem which is solved by the linear system

$$(M_k M_k^T + d_k d_k^T) [y_k^*]_{\mathcal{W}_k} = -\gamma d_k, \quad (10)$$

This can be rewritten as

$$M_k M_k^T [y_k^*]_{\mathcal{W}_k} = -d_k (\gamma + d_k^T [y_k^*]_{\mathcal{W}_k}) = -\sigma_k d_k, \quad (11)$$

where the last equality follows from  $[y_k^*]_{\bar{\mathcal{W}}_k} = 0$  which gives

$$\gamma + d_k^T [y_k^*]_{\mathcal{W}_k} = \gamma + d^T y_k^* = \sigma_k. \quad (12)$$

Our goal is now to formulate a corresponding KKT-system for  $\lambda_k^* \triangleq \frac{y_k^*}{\sigma_k}$  and  $\mu_k \triangleq \frac{w}{\sigma_k}$  instead of  $y_k^*$  and  $w_k$ , respectively. First, to ensure  $\lambda_k^*$  and  $\mu_k$  are well-defined, we ensure that division of  $\sigma_k$  is valid, i.e., that  $\sigma_k$  is nonzero when  $M_k M_k^T$  is nonsingular. Even more strongly,  $\sigma_k$  is positive, as proved by the following lemma.

*Lemma 2:*  $\sigma_k \triangleq \gamma + d^T y_k^* > 0$ , if  $M_k M_k^T$  is nonsingular.

*Proof:* When  $M_k M_k^T$  is nonsingular, (11) gives that

$$[y_k^*]_{\mathcal{W}_k} = -\sigma_k (M_k M_k^T)^{-1} d_k. \quad (13)$$

Inserting this into the definition of  $\sigma_k$  gives

$$\sigma_k = \gamma + d_k^T [y_k^*]_{\mathcal{W}_k} = \gamma - \sigma_k d_k^T (M_k M_k^T)^{-1} d_k \Leftrightarrow$$

$$\sigma_k = \gamma / (1 + d_k^T (M_k M_k^T)^{-1} d_k). \quad (14)$$

By definition,  $\gamma > 0$  and the denominator is nonnegative since  $M_k M_k^T$  is positive definite, resulting in  $\sigma_k > 0$ . ■

Now a KKT-system in terms of  $\lambda_k^*$  and  $\mu_k$ , instead of  $y_k^*$  and  $w_k$ , can be formed by subtracting  $dd^T y_k^*$  from both sides of the first equation of (8) and dividing both sides with  $\sigma_k \neq 0$ , resulting in

$$\begin{pmatrix} MM^T & I_{\bar{\mathcal{W}}_k}^T \\ I_{\bar{\mathcal{W}}_k} & 0 \end{pmatrix} \begin{pmatrix} \lambda_k^* \\ \mu_k \end{pmatrix} = \begin{pmatrix} -d \\ 0 \end{pmatrix}. \quad (15)$$

The solution to the KKT-system in (15) is

$$[\lambda_k^*]_{\mathcal{W}_k} = -(M_k M_k^T)^{-1} d_k, \quad (16a)$$

$$[\mu_k]_{\bar{\mathcal{W}}_k} = [M]_{\bar{\mathcal{W}}_k} M_k^T [\lambda_k^*]_{\mathcal{W}_k} + [d]_{\bar{\mathcal{W}}_k}, \quad (16b)$$

and  $[\lambda_k^*]_{\bar{\mathcal{W}}_k} = 0$ ,  $[\mu_k]_{\mathcal{W}_k} = 0$ .

### B. Checking for Global Optimality and Adding Index to $\mathcal{W}$

In Algorithm 1 the global optimum has been found if

$$w_k \geq -(\gamma + d^T y_k^*) \epsilon. \quad (17)$$

Otherwise, an index corresponding to the most negative component of  $w_k$  is added to  $\mathcal{W}$ , according to Line 9 of Algorithm 1.

The following lemma shows that the dual variable of the KKT-system in (15),  $\mu_k$ , can be considered instead of  $w_k$  when checking for global optimality and for deciding which index that should be added to  $\mathcal{W}$ .

*Lemma 3:* When  $M_k M_k^T$  is nonsingular

$$1) w_k \geq -(\gamma + d^T y_k^*) \epsilon \Leftrightarrow \mu_k \geq -\epsilon.$$

$$2) \operatorname{argmin}_{j \in \bar{\mathcal{W}}_k} [w_k]_j = \operatorname{argmin}_{j \in \bar{\mathcal{W}}_k} [\mu_k]_j.$$

*Proof:* Since  $M_k M_k^T$  is nonsingular,  $\sigma_k > 0$  from Lemma 2. Dividing both sides of  $w_k \geq -(\gamma + d^T y_k^*) \epsilon$  with  $\sigma_k$  proves 1). Furthermore, the positiveness of  $\sigma_k$  gives  $\operatorname{argmin}_{j \in \bar{\mathcal{W}}_k} [w_k]_j / \sigma_k = \operatorname{argmin}_{j \in \bar{\mathcal{W}}_k} [\mu_k]_j$ . ■

### C. Updating Iterate and Removing Component From $\mathcal{W}$

The iterate  $y_k$  is updated in Line 12 of Algorithm 1 according to the line search

$$y_{k+1} = y_k + \alpha_k (y_k^* - y_k), \quad (18)$$

where  $\alpha_k = \min_{h \in \mathcal{W}_k} : [y_k^*]_h < 0 \alpha_k^h$ , with  $\alpha_k^i$  being defined as

$$\alpha_k^i \triangleq [y_k]_i / ([y_k - y_k^*]_i). \quad (19)$$

$\alpha_k^i$  can be interpreted as the step length taken from  $y_k$  in the direction  $y_k - y_k^*$  which makes the  $i$ :th component of  $y$  zero. Also, note that  $[y_k^*]_h < 0 \implies \alpha_k^h \in [0, 1)$ .

Now, we are interested in the corresponding update of  $\lambda$  when  $y$  is updated according to (18). Inserting (18) in the definition of  $\lambda$  in (6) gives

$$\lambda_{k+1} = \frac{y_{k+1}}{\gamma + d^T y_{k+1}} = \frac{y_k + \alpha_k (y_k^* - y_k)}{\gamma + d^T (y_k + \alpha_k (y_k^* - y_k))}. \quad (20)$$

We will now show that the update of  $\lambda_k$  also can be seen as a line search.

*Lemma 4:* If  $M_k M_k^T$  is nonsingular,  $\exists \beta_k \in \mathbb{R}$  such that

$$\lambda_{k+1} = \lambda_k + \beta_k(\lambda_k^* - \lambda_k). \quad (21)$$

*Proof:* The lemma follows from linear fractional transformations conserving convex sets see, e.g., [19, Sec. 2.3.3]. Concretely, picking

$$\beta_k = \frac{\alpha_k \sigma_k}{\alpha_k \sigma_k + (1 - \alpha_k) \rho_k} \quad (22)$$

and inserting it into (21) results in (20) by using (7). ■

Furthermore,  $\beta_k^i$  is defined by (22) when  $\alpha_k^i$  is used instead of  $\alpha_k$ . Before considering properties of  $\beta_k^i$ , we prove that, similar to  $\sigma_k$ ,  $\rho_k > 0$  when  $M_k M_k^T$  is nonsingular.

*Lemma 5:* If  $M_k M_k^T$  is nonsingular,  $\rho_k \triangleq \gamma + d^T y_k > 0$

*Proof:* The lemma is proven by induction. First, inserting  $y_{k+1}$  from (18) into the definition of  $\rho$  in (7) gives

$$\begin{aligned} \rho_{k+1} &= \gamma + d^T (y_k + \alpha_k (y_k^* - y_k)) \\ &= \gamma + d^T y_k + \alpha_k (d^T y_k^* - d^T y_k + \gamma - \gamma) \\ &= (1 - \alpha_k) \rho_k + \alpha_k \sigma_k. \end{aligned} \quad (23)$$

Now, assume that  $\rho_k > 0$ . Then  $\rho_{k+1} > 0$  since  $\alpha_k \in [0, 1]$  and  $\sigma_k > 0$  from Lemma 2. The base case is satisfied since  $y_1 = 0 \implies \rho_1 = \gamma > 0$ . Hence, the lemma follows by induction. ■

This nonnegativity property of  $\rho_k$ , together with the nonnegativity property of  $\sigma_k$ , can be used to prove the following lemma which establishes a relation between  $\alpha_k^i$  and  $\beta_k^i$ .

*Lemma 6:* If  $M_k M_k^T$  is nonsingular and  $\alpha_k^i, \alpha_k^j \in [0, 1]$ ,

$$\alpha_k^i \leq \alpha_k^j \Leftrightarrow \beta_k^i \leq \beta_k^j. \quad (24)$$

*Proof:* Directly using the definition of  $\beta_k^i$  from (22), and dropping the subscript  $k$  for convenience, gives

$$\begin{aligned} \beta^i \leq \beta^j &\Leftrightarrow \frac{\alpha^i \sigma}{\alpha^i \sigma + (1 - \alpha^i) \rho} \leq \frac{\alpha^j \sigma}{\alpha^j \sigma + (1 - \alpha^j) \rho} \\ &\Leftrightarrow \alpha^i \sigma \rho \leq \alpha^j \sigma \rho \Leftrightarrow \alpha^i \leq \alpha^j, \end{aligned} \quad (25)$$

where the nonnegativeness of  $\rho_k$  and  $\sigma_k$  has been used in the second and third equivalence. ■

We are now ready to state the main result for the nonsingular case. The following lemma shows that  $\lambda_k^*$  can be considered instead of  $y_k^*$  when checking for local optimality and for deciding which index that should be removed from  $\mathcal{W}$ .

*Lemma 7:* If  $M_k M_k^T$  is nonsingular

- 1)  $y_k^* \geq 0 \Leftrightarrow \lambda_k^* \geq 0$ .
- 2)  $\operatorname{argmin}_{h \in \mathcal{W}_k : [y_k^*]_h < 0} \alpha_k^h = \operatorname{argmin}_{h \in \mathcal{W}_k : [\lambda_k^*]_h < 0} \beta_k^h$ .

*Proof:* First, since  $M_k M_k^T$  is nonsingular we have that  $\sigma_k > 0$  which gives

$$\mathcal{H}_k \triangleq \{h \in \mathcal{W}_k : [y_k^*]_h < 0\} = \{h \in \mathcal{W}_k : [\lambda_k^*]_h < 0\},$$

since  $\frac{y_k^*}{\sigma_k} = \lambda_k^*$ , i.e., the same indices of  $y_k^*$  and  $\lambda_k^*$  will be negative, and these components are given by the set  $\mathcal{H}_k$ , which proves 1). Next,  $[y_k^*]_h < 0$  inserted into (19) gives  $\alpha_k^h \in [0, 1]$ ,  $\forall h \in \mathcal{H}_k$ . Hence, Lemma 6 gives the same ordering of  $\{\alpha_k^h\}_{h \in \mathcal{H}_k}$  and  $\{\beta_k^h\}_{h \in \mathcal{H}_k}$  which means that the same index will give a minimum. ■

## D. Singular Case

$M_k M_k^T$  only becomes singular after a component is added to  $\mathcal{W}$  in Algorithm 1. In this case, the solution to (5),  $[y_k^*]_{\mathcal{W}_k}$ , will be a singular eigenvector to  $M_k M_k^T$  as is shown by the following lemma.

*Lemma 8:* If  $M_k M_k^T$  becomes singular in Algorithm 1,  $M_k M_k^T [y_k^*]_{\mathcal{W}_k} = 0$  and  $\sigma_k = 0$ .

*Proof:* If  $M_k M_k^T$  is singular,  $\exists \tilde{\lambda}_k \neq 0, \tilde{\lambda}_k \in \mathbb{R}^m$  such that  $M_k M_k^T [\tilde{\lambda}_k]_{\mathcal{W}_k} = 0, [\tilde{\lambda}_k]_{\tilde{\mathcal{W}}_k} = 0$ . Now, define  $\delta_k \triangleq d^T \tilde{\lambda}_k$ . Then  $y_k^* = -\frac{\gamma}{\delta_k} \tilde{\lambda}_k$  leads to the objective function of (5) being zero and hence, since norms are nonnegative, this is a minimizer of (5). Inserting this  $y_k^*$  into the definition of  $\sigma_k$  gives  $\sigma_k = 0$  by construction.

What remains to prove is that  $\delta_k \neq 0$ , so  $y_k^*$  from above is well-defined. Since  $M_k M_k^T$  only becomes singular after an addition to  $\mathcal{W}$ , let  $i$  be the component that was added to  $\mathcal{W}$  at iteration  $k - 1$ , i.e.,  $\mathcal{W}_k = \mathcal{W}_{k-1} \cup \{i\}$ . From (15),  $\mu_{k-1}$  is given as

$$MM_{k-1}^T \lambda_{k-1}^* + d = \mu_{k-1}. \quad (26)$$

Multiplying this equation with  $\tilde{\lambda}_k^T$  from the left gives

$$\tilde{\lambda}_k^T MM_{k-1}^T \lambda_{k-1}^* + \tilde{\lambda}_k^T d = [\tilde{\lambda}_k]_i [\mu_{k-1}]_i \Leftrightarrow \quad (27a)$$

$$\tilde{\lambda}_k^T d = [\tilde{\lambda}_k]_i [\mu_{k-1}]_i, \quad (27b)$$

where we have recalled the partitions of  $\lambda^*$  and  $\mu$  from (16a) and (16b), respectively. Furthermore, we have also used that  $\tilde{\lambda}_k$  is a singular eigenvector of  $M^T$  by construction.

Since  $i$  was added to  $\mathcal{W}$  at iteration  $k - 1$ ,  $[\mu_{k-1}]_i < 0$ . Furthermore,  $[\tilde{\lambda}_k]_i \neq 0$  since  $[\tilde{\lambda}_k]_{\tilde{\mathcal{W}}_k}^T M_k = 0$  and  $[\tilde{\lambda}_k]_i = 0$  would imply that  $[\tilde{\lambda}_k]_{\tilde{\mathcal{W}}_{k-1}}^T M_{k-1} = 0$ , which is impossible since  $M_{k-1} M_{k-1}^T$  was nonsingular, hence,  $\delta_k \triangleq \tilde{\lambda}_k^T d \neq 0$ . ■

*Remark 3:* We will, without loss of generality, assume that  $\tilde{\lambda}_k$  is such that  $\delta < 0$ . This is valid since  $M_k M_k^T \tilde{\lambda}_k = 0 \implies M_k M_k^T (-\tilde{\lambda}_k) = 0$ . Hence, we can always change the sign of  $\delta_k$  by changing the sign of  $\tilde{\lambda}_k$ .

Using Lemma 8 together with (20) gives the following update of  $\lambda$  in the singular case

$$\begin{aligned} \lambda_{k+1} &= \frac{y_k + \alpha_k (y_k^* - y_k)}{\gamma + d^T (y_k + \alpha_k (y_k^* - y_k))} \\ &= \frac{(1 - \alpha_k) y_k + \alpha_k y_k^*}{(1 - \alpha_k) \rho_k} = \lambda_k + \frac{\alpha_k}{(1 - \alpha_k) \rho_k} y_k^* \\ &= \lambda_k - \frac{\alpha_k \gamma}{(1 - \alpha_k) \rho_k \delta_k} \tilde{\lambda}_k = \lambda_k + \tilde{\beta}_k \tilde{\lambda}_k, \end{aligned} \quad (28)$$

where  $\sigma_k = 0$  is used in the second equality,  $y_k^* = -\frac{\gamma}{\delta_k} \tilde{\lambda}_k$  is used in the fourth equality and  $\tilde{\beta}_k \triangleq -\frac{\alpha_k \gamma}{(1 - \alpha_k) \rho_k \delta_k}$  has been defined in the last equality. Similar to  $\alpha_k^i$  and  $\beta_k^i$ , we introduce the definition

$$\tilde{\beta}_k^i \triangleq -\frac{\gamma}{\rho_k \delta_k} \cdot \frac{\alpha_k^i}{(1 - \alpha_k^i)} = -\frac{[\lambda_k]_i}{[\tilde{\lambda}_k]_i} \quad (29)$$

to denote the step length which results in  $[\lambda_{k+1}]_i = 0$  when a step is taken in the direction  $\tilde{\lambda}_k$  during iteration  $k$ .

*Remark 4:* The definition of  $\tilde{\beta}_k^i$  in (29) is well-defined since  $\rho_k > 0, \delta_k < 0$  and  $\alpha_k^i \in [0, 1]$ .  $\delta_k < 0$  has been established in Lemma 8,  $\rho_k > 0$  follows from  $M_k M_k^T$  only becoming

singular after a constraint has been added to  $\mathcal{W}_k$ , which implies that  $y_k = y_{k-1}^* \implies \rho_k = \sigma_{k-1} > 0$  since  $M_{k-1}M_{k-1}^T$  was nonsingular.

Analogous to Lemma 6 for the nonsingular case, we establish the following properties for  $\tilde{\beta}_k^i$

*Lemma 9:* If  $\alpha_k^i, \alpha_k^j \in [0, 1)$  then

$$\alpha_k^j \leq \alpha_k^i \Leftrightarrow \tilde{\beta}_k^j \leq \tilde{\beta}_k^i \quad (30)$$

*Proof:* Using the definition of  $\tilde{\beta}_k^i$  in (29)

$$\begin{aligned} \tilde{\beta}_k^j \leq \tilde{\beta}_k^i &\Leftrightarrow -\frac{\gamma}{\rho_k \delta_k} \cdot \frac{\alpha_k^j}{(1 - \alpha_k^j)} \leq -\frac{\gamma}{\rho_k \delta_k} \cdot \frac{\alpha_k^i}{(1 - \alpha_k^i)} \\ &\Leftrightarrow (1 - \alpha_k^i) \alpha_k^j \leq (1 - \alpha_k^j) \alpha_k^i \Leftrightarrow \alpha_k^j \leq \alpha_k^i, \end{aligned} \quad (31)$$

where  $-\frac{\gamma}{\rho_k \delta_k} > 0$  and  $\alpha_k^i, \alpha_k^j \in [0, 1)$  have been used in the second equivalence.  $-\frac{\gamma}{\rho_k \delta_k} > 0$  follows from Remark 4. ■

The following lemma is analogous to Lemma 7 but for the singular case. It shows that  $\tilde{\lambda}_k$  can be considered instead of  $y_k^*$  when removing indices from  $\mathcal{W}$  in the singular case.

*Lemma 10:* If  $M_k M_k^T$  is singular

- 1)  $y_k^* \geq 0 \Leftrightarrow \tilde{\lambda}_k \geq 0$ .
- 2)  $\operatorname{argmin}_{h \in \mathcal{W}_k : [y_k^*]_h < 0} \alpha_k^h = \operatorname{argmin}_{h \in \mathcal{W}_k : [\tilde{\lambda}_k]_h < 0} \tilde{\beta}_k^h$ .

*Proof:* If  $M_k M_k^T$  is singular we have from Lemma 8 that  $y_k^* = -\frac{\gamma}{\delta_k} \tilde{\lambda}_k$ , hence,

$$\mathcal{H}_k \triangleq \{h \in \mathcal{W}_k : [y_k^*]_h < 0\} = \{h \in \mathcal{W}_k : [\tilde{\lambda}_k]_h < 0\},$$

since  $-\frac{\gamma}{\delta_k} > 0$ , i.e., the same indices of  $y_k^*$  and  $\tilde{\lambda}_k$  will be negative, and these components are given by the set  $\mathcal{H}_k$ , which proves 1). Next,  $[y_k^*]_h < 0$  inserted into (19) gives  $\alpha_k^h \in [0, 1), \forall h \in \mathcal{H}_k$ , hence, Lemma 9 can be used and gives the same ordering of  $\{\alpha_k^h\}_{h \in \mathcal{H}_k}$  and  $\{\tilde{\beta}_k^h\}_{h \in \mathcal{H}_k}$  which means that the same index will give a minimum. ■

*Remark 5:* From Farkas' lemma it is necessary for at least one component of  $y_k^*$ , and hence of  $\tilde{\lambda}_k$ , to be negative if the QP is feasible, see [11, Th. 1]. Therefore, since we assume that (1) is feasible, at least one constraint will be removed from  $\mathcal{W}_k$  at iteration  $k$  if  $M_k M_k^T$  is singular, regaining nonsingularity of  $M_{k+1} M_{k+1}^T$ .

#### IV. CERTIFICATION OF NNLS ALGORITHM

We will now use the properties of Algorithm 1, which have been established in Section III, to certify its iteration complexity for all parameters in  $\Theta_0$ . This is done by proving that Algorithm 1 produces the same working-set sequence as a standard primal active-set method, e.g., [7], [20], [21], applied to the dual of (1).

After this equivalence has been established, the result from [15] can be used to determine the working-set changes and the number of iterations any  $\theta \in \Theta_0$  produces, which is done by applying the certification method presented in [15] to the dual of (1). The dual problem of (1), using the definitions of  $M$  and  $d(\theta)$  from (2), is given by

$$\underset{\lambda \geq 0}{\text{minimize}} \quad \frac{1}{2} \lambda^T M M^T \lambda + d^T(\theta) \lambda, \quad (32)$$

**Algorithm 2** A Standard Primal Active-Set Quadratic Programming Method Applied to (32) [15]

---

```

1:  $v \leftarrow \mathcal{L}^{-T}(f_\theta \theta + f)$ ;  $d \leftarrow (b + W\theta) + Mv$ 
2:  $k \leftarrow 1, \mathcal{W}_k \leftarrow \emptyset; \lambda_k \leftarrow 0$ ;
3: while true do
4:   if  $M_k M_k^T$  is singular then go to step 16
5:    $\lambda_k^* \leftarrow$  solution to KKT-system (15)
6:   if  $\lambda_k^* \geq 0$  then
7:      $\mu_k \leftarrow M(M_k^T[\lambda_k^*]_{\mathcal{W}_k}) + d$ 
8:     if  $\mu_k \geq -\epsilon$  then
9:       return  $\lambda^*, x^* \leftarrow -\mathcal{L}^{-1}(M_k^T \lambda_k^* + v), \mathcal{W}_k$ 
10:    else  $i \leftarrow \operatorname{arg min} [\mu_k]_i; \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \cup \{i\}$ 
11:         $\lambda_{k+1} \leftarrow \lambda_k^*$ 
12:    else  $l \leftarrow \operatorname{arg min}_{h \in \mathcal{W}_k : [\lambda_k^*]_h < 0} \left\{ \frac{[\lambda_k]_h}{[\lambda_k - \lambda_k^*]_h} \right\}; \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{l\}$ 
13:         $\beta_k \leftarrow \frac{[\lambda_k]_l}{[\lambda_k - \lambda_k^*]_l}; \lambda_{k+1} \leftarrow \lambda_k + \beta_k(\lambda_k^* - \lambda_k)$ 
14:     $k \leftarrow k + 1$ 
15:  end while
16:  $\tilde{\lambda}_k \leftarrow$  solution to  $M_k M_k^T \tilde{\lambda}_k = 0, d_k^T \tilde{\lambda}_k < 0$ 
17: if  $\tilde{\lambda}_k \geq 0$  then
18:   return primal infeasible
19: else  $l \leftarrow \operatorname{arg min}_{h \in \mathcal{W}_k : [\tilde{\lambda}_k]_h < 0} \left\{ -\frac{[\tilde{\lambda}_k]_h}{[\tilde{\lambda}_k]_h} \right\}; \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{l\}$ 
20:    $\tilde{\beta}_k \leftarrow -\frac{[\tilde{\lambda}_k]_l}{[\tilde{\lambda}_k]_l}; \lambda_{k+1} \leftarrow \lambda_k + \tilde{\beta}_k \tilde{\lambda}_k$ 
21:    $k \leftarrow k + 1$ 
22: go to step 5

```

---

Algorithm 2 presents a standard primal active-set method which is applied to solve (32). A complexity certification method for Algorithm 2, which determines the working-set sequences that are produced by Algorithm 2 for every  $\theta \in \Theta_0$ , is presented in [15]. We are now ready to state the main result of this letter, namely that Algorithm 1 will produce the same working-set sequence as Algorithm 2, for which there exists a certification method to determine *exactly* which working-set sequence any parameter will generate [15].

*Theorem 1:* Let  $\tilde{k}(\theta)$  be the number of iterations needed by Algorithm 2 to terminate and let  $\{\tilde{\mathcal{W}}_k(\theta)\}_{k=1}^{\tilde{k}(\theta)}$  be the corresponding working-set sequence produced by Algorithm 2. Then Algorithm 1 terminates in  $\tilde{k}(\theta)$  iterations and produces the working-set sequence  $\{\tilde{\mathcal{W}}_k(\theta)\}_{k=1}^{\tilde{k}(\theta)}, \forall \theta \in \Theta_0$ .

*Proof:* The theorem will be proven by using the properties derived in Section III to map an iteration of Algorithm 1 to an iteration of Algorithm 2. Table I summarizes the correspondence between each line of Algorithm 1 to lines of Algorithm 2 and which equation, Lemma or Remark proves each correspondence. Both the case when  $M_k M_k^T$  is singular and nonsingular is shown in Table I. ■

A direct consequence of Theorem 1 is that the worst-case number of iterations when (1) is solved with Algorithm 1 can be certified by applying the complexity certification method from [15] to the dual mpQP given in (32).

*Remark 6:* There are numerous active-set algorithms that are equivalent, see [22]. As is discussed in [15], Algorithm 2

TABLE I  
MAPPING FROM ALGORITHM 1 TO ALGORITHM 2

Alg. 1	Alg. 2	nonsingular	Alg. 2	singular
#1-3	#1-3	-	#1-3	-
#4	#5	(15)	#16	Lemma 8
#5	#6	Lemma 7	#17	Lemma 10
#6-10	#7-11	Lemma 3	#18	Remark 5
#11-13	#12-13	Lemma 7	#19-20	Lemma 10
#14	#14	-	#21	-
#15	#9	(6)	Impossible if primal feasible	

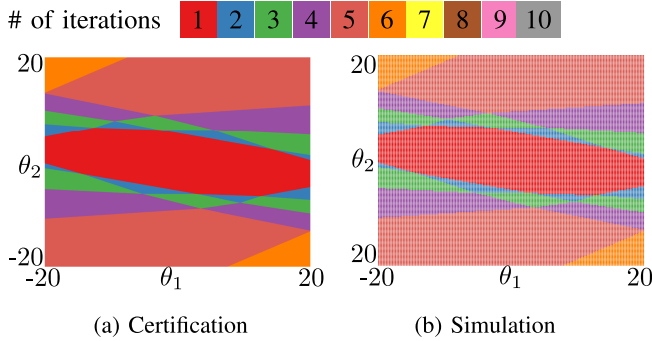


Fig. 1. Number of iterations determined by: (a) Applying the certification method presented in [15] to the dual mpQP; (b) Executing Algorithm 1 over a two-dimensional grid in the parameter space.  $\theta_i = 0$  for  $i \neq 1, 2$ .

is equivalent to, e.g., Dantzig’s active-set method for QPs [21] applied to the dual QP. Theorem 1 together with this equivalence explain the empirical observation in [11] that Algorithm 1 produces the same number of iterations as Dantzig’s method.

## V. NUMERICAL EXAMPLE

To exemplify the complexity certification method for Algorithm 1, an mpQP originating from the application of MPC to an inverted pendulum was considered. The resulting mpQP had the dimensions,  $m = 10$ ,  $p = 8$ , and  $n = 5$ . Further details about the problem are given in [14].

The certification method was compared with results obtained by drawing samples from  $\Theta_0$  and executing Algorithm 1 on the resulting QPs. Figure 1 compares such simulations of Algorithm 1 for  $\theta$  taken on a grid on a 2-dimensional subspace of  $\Theta_0$ , with a slice of the partition, corresponding to the same subspace, obtained by applying the certification method from [15] to the dual of the mpQP. As Theorem 1 predicts, the resulting number of iterations is equal for the simulation and the certification. In addition,  $10^8$  random samples of the entire  $\Theta_0$  were taken and Algorithm 1 was applied to the resulting mpQPs. As before, these simulation results were compared with the results from applying the certification method from [15] to the dual of the mpQP. Again, as predicted by Theorem 1, both the simulation and the certification resulted in the same number of iterations.

The computation time required for the complexity certification of the inverted pendulum example was 7.6 seconds when executed on an Intel 2.7 GHz i7-7500U CPU. For more details about the certification method itself, such as complexity, see [15].

## VI. CONCLUSION

This letter has proposed a complexity certification method for a simple and efficient QP method. The complexity certification was done by relating the QP method to a standard primal active-set method applied to the dual of the QP, allowing the complexity certification method in [15] to be directly applicable. Future research includes certifying the complexity of the extended method presented in [17], which improves the numerical stability of the QP method considered in this letter.

## REFERENCES

- [1] C. V. Rao, S. J. Wright, and J. B. Rawlings, “Application of interior-point methods to model predictive control,” *J. Optim. Theory Appl.*, vol. 99, no. 3, pp. 723–757, 1998.
- [2] Y. Wang and S. Boyd, “Fast model predictive control using online optimization,” *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 2, pp. 267–278, Mar. 2010.
- [3] D. Axehill and A. Hansson, “A dual gradient projection quadratic programming algorithm tailored for model predictive control,” in *Proc. 47th IEEE Conf. Decis. Control*, Dec. 2008, pp. 3057–3064.
- [4] P. Patrinos and A. Bemporad, “An accelerated dual gradient-projection algorithm for embedded linear model predictive control,” *IEEE Trans. Autom. Control*, vol. 59, no. 1, pp. 18–33, Jan. 2014.
- [5] S. Richter, C. N. Jones, and M. Morari, “Computational complexity certification for real-time MPC with input constraints based on the fast gradient method,” *IEEE Trans. Autom. Control*, vol. 57, no. 6, pp. 1391–1403, Jun. 2012.
- [6] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, “OSQP: An operator splitting solver for quadratic programs,” Nov. 2017. [Online]. Available: arXiv:1711.08013.
- [7] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, NY, USA: Springer, 2006.
- [8] D. Goldfarb and A. Idnani, “A numerically stable dual method for solving strictly convex quadratic programs,” *Math. Program.*, vol. 27, pp. 1–33, Sep. 1983.
- [9] H. J. Ferreau, H. G. Bock, and M. Diehl, “An online active set strategy to overcome the limitations of explicit MPC,” *Int. J. Robust Nonlinear Control*, vol. 18, no. 8, pp. 816–830, 2008.
- [10] K. Kunisch and F. Rendl, “An infeasible active set method for quadratic problems with simple bounds,” *SIAM J. Optim.*, vol. 14, no. 1, pp. 35–52, Jan. 2003.
- [11] A. Bemporad, “A quadratic programming algorithm based on nonnegative least squares with applications to embedded model predictive control,” *IEEE Trans. Autom. Control*, vol. 61, no. 4, pp. 1111–1116, Apr. 2016.
- [12] D. Arnström and D. Axehill, “Exact complexity certification of a standard primal active-set method for quadratic programming,” in *Proc. IEEE 58th Conf. Decis. Control*, Dec. 2019, pp. 4317–4324.
- [13] G. Cimini and A. Bemporad, “Exact complexity certification of active-set methods for quadratic programming,” *IEEE Trans. Autom. Control*, vol. 62, no. 12, pp. 6094–6109, Dec. 2017.
- [14] G. Cimini and A. Bemporad, “Complexity and convergence certification of a block principal pivoting method for box-constrained quadratic programs,” *Automatica*, vol. 100, pp. 29–37, Feb. 2019.
- [15] D. Arnström and D. Axehill, “A unifying complexity certification framework for active-set methods for convex quadratic programming,” Mar. 2020. [Online]. Available: arXiv:2003.07605.
- [16] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, “The explicit linear quadratic regulator for constrained systems,” *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.
- [17] A. Bemporad, “A numerically stable solver for positive semidefinite quadratic programs based on nonnegative least squares,” *IEEE Trans. Autom. Control*, vol. 63, no. 2, pp. 525–531, Feb. 2018.
- [18] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, vol. 15. Philadelphia, PA, USA: SIAM, 1995.
- [19] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
- [20] R. Fletcher, “A general quadratic programming algorithm,” *IMA J. Appl. Math.*, vol. 7, no. 1, pp. 76–91, 1971.
- [21] G. B. Dantzig, *Linear Programming and Extensions*, vol. 48. Princeton, NJ, USA: Princeton Univ. Press, 1998.
- [22] M. J. Best, “Equivalence of some quadratic programming algorithms,” *Math. Program.*, vol. 30, no. 1, p. 71, 1984.