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

Daniel Arnström, Alberto Bemporad, Fellow, IEEE, and Daniel Axehill

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

A N 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

$$\minimize_{x} \frac{1}{2} x^T H x + (f^T + \theta^T f_0^T) x$$
subject to $Ax \leq b + W \theta,$

(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 $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $W \in \mathbb{R}^{m \times p}$, $f \in \mathbb{R}^n$, $f_0 \in \mathbb{R}^{n \times p}$, and $H \in \mathbb{S}^n_{++}$. The minimizer of (1), given $\theta$, is denoted $x^*(\theta)$. It is well-known
Algorithm 1: For a more detailed description, see [11] or [18, 23].

B. Nonnegative Least-Squares Method

is also introduced for convenience.

The nonnegative least-squares method for solving (1) is presented below.

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

1: \( v \leftarrow \mathcal{L}^{-T}(f_0 \theta + f) \); \( d \leftarrow (b + W \theta) + Mv \)
2: \( k \leftarrow 1 \); \( \mathcal{W}_k \leftarrow \theta \); \( y_k \leftarrow 0 \)
3: while true do
4: \( y_k^* \leftarrow \text{solution to least squares problem (5)} \)
5: if \( y_k^* \geq 0 \) then
6: \( w_k \leftarrow (M^T[M^T y_k^* \theta \theta] + (\gamma + d_1^T y_k^* \theta \theta)]d \)
7: if \( w_k \geq (\gamma + d_1^T y_k^* \theta \theta) e \) then
8: go to step 15
9: else \( i \leftarrow \arg \min \{ w_k \} \), \( \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \cup \{ i \} \)
10: \( y_{k+1} \leftarrow y_k^* \)
11: else \( l \leftarrow \arg \min \{ \frac{|y_k^h|}{\gamma y_k^h - \beta^T y_k} \} \)
12: \( \alpha_k \leftarrow \frac{|y_k^h|}{\gamma y_k^h - \beta^T y_k} ; \ y_{k+1} \leftarrow y_k + \alpha_k (y_k^* - y_k) \)
13: \( \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{ l \} \)
14: \( k \leftarrow k + 1 \)
15: return \( x^* \leftarrow \frac{y_k^*}{\gamma + d_2^T y_k} \), \( x^* \leftarrow \mathcal{L}^{-1}(M^T \theta \theta^* + v) , \mathcal{W}_k \)

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

\[
\text{minimize } y \left\{ \frac{1}{2} \left[ M^T \left( d(\theta)^T \right) y + \left[ 0 \right] \right] \right\}_2 \\
\text{subject to } [y]_i = 0, \quad i \in \mathcal{W}_k
\]

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 all \( y_k^* \geq 0 \), 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 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 extend 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 PQ (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 \( X_i = \{ x \in \mathbb{R}^n : Ax \leq b + W \theta_i \} \) be the feasible set for (1) when \( \theta = \theta_i \). Then problem (1) is feasible \( \forall \theta \in \Theta_0 \iff X_i \neq \emptyset, \forall i \in \{ 1, \ldots, M \} \).

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

$$
\lambda \triangleq \frac{y}{y + d^T y}.
$$

(6)

The following scalars will also prove useful

$$
\sigma_k \triangleq y + d^T y_k^*, \quad \rho_k \triangleq y + d^T y_k.
$$

(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 $MK_{k}^T$ 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

$$
\begin{pmatrix}
MM^T + dd^T & [I_{W_k}^T] \\
W_k & 0
\end{pmatrix}
\begin{bmatrix}
y_k^* \\
w_k
\end{bmatrix}
= \begin{bmatrix}
-\gamma d \\
0
\end{bmatrix},
$$

(8)

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

$$
y_k^* = -\gamma I_k^T ((MK_k^T + dd_k^T)^{-1} d_k),
$$

(9)

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

$$
(MK_k^T + dd_k^T) [y_k^*]_{\bar{W}_k} = -\gamma d_k.
$$

(10)

This can be rewritten as

$$
MK_k^T [y_k^*]_{\bar{W}_k} = -d_k (y + d^T y_k^*)^{-1} d_k,
$$

(11)

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

$$
\gamma + d^T y_k^* = \sigma_k.
$$

(12)

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

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

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

$$
[y_k^*]_{\bar{W}_k} = -\sigma_k (MK_k^T)^{-1} d_k.
$$

(13)

Inserting this into the definition of $\sigma_k$ gives

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

(14)

By definition, $\gamma > 0$ and the denominator is nonnegative since
$MK_k^T$ 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 $dd_k^T y_k^*$ from both sides of the first equation of (8) and dividing both sides with $\sigma_k \neq 0$, resulting in

$$
\begin{pmatrix}
MM^T & [I_{W_k}^T] \\
W_k & 0
\end{pmatrix}
\begin{bmatrix}
\lambda_k^* \\
\mu_k
\end{bmatrix}
= \begin{bmatrix}
-\gamma d \\
0
\end{bmatrix}.
$$

(15)

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

$$
[\lambda_k^*]_{\bar{W}_k} = -(MK_k^T)^{-1} d_k,
$$

(16a)

$$
[\mu_k]_{\bar{W}_k} = [M]_{\bar{W}_k} [\lambda_k^*]_{\bar{W}_k} + [d]_{\bar{W}_k}.
$$

(16b)

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

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

In Algorithm 1 the global optimum has been found if

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

(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 $MK_k^T$ is nonsingular

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

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

Proof: Since $MK_k^T$ 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 $\text{argmin}_{j \in \mathcal{W}_k} [w_k]_j / \sigma_k = \text{argmin}_{j \in \mathcal{W}_k} [\mu_k]_j$.

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

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

(18)

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

$$
\alpha_k \triangleq [y_k^*]_h / ([y_k]_h - [y_k^*]_h).
$$

(19)

$\alpha_k$ can be interpreted as the step length taken from $y_k$ in the
direction $y_k^* - y_k$ which makes the $h$th component of $y$ zero.

Also, note that $[y_k^*]_h < 0 \implies \alpha_k \in [0, 1]$.

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 (16) gives

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

(20)

We will now show that the update of $\lambda$ also can be seen as a
line search.
Lemma 4: If $M_kM_k^T$ is nonsingular, $\exists \beta_k \in \mathbb{R}$ such that
\begin{equation}
\lambda_{k+1} = \lambda_k + \beta_k(\lambda_k^* - \lambda_k).
\end{equation}

Proof: The lemma follows from linear fractional transformations conserving convex sets see, e.g., [19, Sec. 2.3.3]. Concretely, picking
\begin{equation}
\beta_k = \frac{\alpha_k \sigma_k}{\alpha_k \sigma_k + (1 - \alpha_k) \rho_k}
\end{equation}
and inserting it into (21) results in (20) by using (7).

Furthermore, $\beta_k^*$ is defined by (22) when $\alpha_k^* \gamma + d^T y_k > 0$. Before considering properties of $\beta_k^*$, we prove that, similar to $\sigma_k$, $\rho_k > 0$ when $M_kM_k^T$ is nonsingular.

Lemma 5: If $M_kM_k^T$ is nonsingular, $\rho_k \equiv \gamma + d^T y_k > 0$

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

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 $\alpha_k^*$ and $\beta_k^*$.

Lemma 6: If $M_kM_k^T$ is nonsingular and $\alpha_k, \beta_k \in [0, 1],$
\begin{equation}
\alpha_k^* \leq \alpha_k \iff \beta_k^* \leq \beta_k.
\end{equation}

Proof: Directly using the definition of $\beta_k^*$ from (22), and dropping the subscript $k$ for convenience, gives
\begin{align*}
\beta^* \leq \beta \iff & \frac{\alpha^* \sigma}{\alpha^* \sigma + (1 - \alpha^*) \rho} \leq \frac{\alpha \sigma}{\alpha \sigma + (1 - \alpha) \rho} \\
& \iff \alpha^* \sigma \rho \leq \alpha \sigma \rho \iff \alpha^* \leq \alpha.
\end{align*}

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 $M_kM_k^T$ is nonsingular
1) $y_k^* \geq 0 \iff \lambda_k^* \geq 0.$
2) argmin $\alpha_k^* \argmin_{h \in \mathcal{W}_k} [y_k^*]_h < 0, \argmin_{h \in \mathcal{W}_k} [\lambda_k^*]_h < 0.$

Proof: First, since $M_kM_k^T$ is nonsingular we have that $\sigma_k > 0$ which gives
\begin{equation}
\mathcal{H}_k \triangleq \{ h \in \mathcal{W}_k : [y_k^*]_h < 0 \} = \{ h \in \mathcal{W}_k : [\lambda_k^*]_h < 0 \},
\end{equation}

since $\lambda_k^* \equiv \lambda_k^*$, 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 $\alpha_k^* \in [0, 1], \forall h \in \mathcal{H}_k$. Hence, Lemma 6 gives the same ordering of $[\alpha_k^*]_{h \in \mathcal{H}_k}$ and $[\beta_k^*]_{h \in \mathcal{H}_k}$ which means that the same index will give a minimum.

D. Singular Case

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

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

Proof: If $M_kM_k^T$ is singular, $\exists \lambda_k \neq 0, \tilde{\lambda}_k \in \mathbb{R}^m$ such that $M_kM_k^T[\tilde{\lambda}_k]_{\mathcal{W}} = 0, [\tilde{\lambda}_k]_{\mathcal{W}} = 0$. Now, define $\delta_k \equiv d^T \tilde{\lambda}_k$. Then $y_k^* = -\frac{\delta_k}{\alpha_k} \tilde{\lambda}_k$ 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 $M_kM_k^T$ 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
\begin{equation}
M_kM_k^T \mu_{k-1}^* + d = \mu_{k-1}.
\end{equation}

Multiplying this equation with $\tilde{\lambda}_k^*$ from the left gives
\begin{equation}
\tilde{\lambda}_k^* M_kM_k^T \mu_{k-1}^* + \tilde{\lambda}_k^* d = [\tilde{\lambda}_k]\mu_{k-1},
\end{equation}

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 $[\tilde{\lambda}_k]_i M_k^i = 0$ and $[\tilde{\lambda}_k]_i = 0$ would imply that $[\tilde{\lambda}_k]^T M_k^T M_k \tilde{\lambda}_k = 0$, which is impossible since $M_kM_k^T \mu_{k-1}^*$ was nonsingular, hence, $\delta_k \neq \tilde{\lambda}_k d \neq 0$.

Remark 3: We will, without loss of generality, assume that $\lambda_k$ is such that $\delta_k < 0$. This is valid since $M_kM_k^T \lambda_k = 0 \implies M_kM_k^T(-\lambda_k) = 0$. Hence, we can always change the sign of $\delta_k$ by changing the sign of $\lambda_k$.

Using Lemma 8 together with (20) gives the following update of $\lambda$ in the singular case
\begin{align*}
\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 + \alpha_k \sigma_k} \\
&= \lambda_k - \frac{\alpha_k}{(1 - \alpha_k)\rho_k} \tilde{\lambda}_k = \lambda_k - \tilde{\beta}_k \tilde{\lambda}_k.
\end{align*}

where $\sigma_k = 0$ is used in the second equality. $y_k^* = -\frac{\gamma}{\alpha_k} \tilde{\lambda}_k$ is used in the fourth equality and $\tilde{\beta}_k \equiv \frac{\alpha_k}{(1 - \alpha_k)\rho_k} \tilde{\lambda}_k$ has been defined in the last equality. Similar to $\alpha_k^*$ and $\beta_k^*$, we introduce the definition
\begin{equation}
\tilde{\beta}_k^* \equiv -\gamma \cdot \frac{\alpha_k}{(1 - \alpha_k)\rho_k} \frac{\lambda_k^*}{[\lambda_k]^*}.
\end{equation}

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 $\tilde{\beta}_k^*$ in (29) is well-defined since $\rho_k > 0, \delta_k < 0$ and $\alpha_k^* \in [0, 1]$. $\delta_k < 0$ has been established in Lemma 8, $\rho_k > 0$ follows from $M_kM_k^T$ only becoming
singular after a constraint has been added to \( \mathcal{W}_k \), which implies that \( y_k = y^*_k \implies \rho_k = \sigma_{k-1} > 0 \) since \( M_{k-1}M^T_{k-1} \) was nonsingular.

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

**Lemma 9:** If \( \alpha^i_k, \alpha^h_k \in [0, 1) \) then
\[
\alpha^i_k \leq \alpha^h_k \implies \tilde{\beta}_k^i \leq \tilde{\beta}_k^h
\]

**Proof:** Using the definition of \( \tilde{\beta}_k^i \) in (29)
\[
\tilde{\beta}_k^i \leq \tilde{\beta}_k^h \iff -\frac{\gamma}{\rho_k \delta_k} \cdot \alpha^i_k (1 - \alpha^i_k) \leq -\frac{\gamma}{\rho_k \delta_k} \cdot \alpha^h_k (1 - \alpha^h_k)
\]
\[
(1 - \alpha^i_k)\alpha^i_k \leq (1 - \alpha^h_k)\alpha^h_k \implies \alpha^i_k \leq \alpha^h_k.
\]

where \( -\frac{\gamma}{\rho_k \delta_k} > 0 \) and \( \alpha^i_k, \alpha^h_k \in [0, 1) \) have been used in the second equivalence. \( -\frac{\gamma}{\rho_k \delta_k} \geq 0 \) 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 \( y_k^\ast \) when removing indices from \( \mathcal{W} \) in the singular case.

**Lemma 10:** If \( M_kM^T_k \) is singular
\[
1) \ y_k^\ast \geq 0 \implies \tilde{\lambda}_k \geq 0;
\]
\[
2) \ \text{argmin} \ \alpha^h_k = \text{argmin} \ \tilde{\beta}_k^h.
\]

**Proof:** If \( M_kM^T_k \) is singular we have from Lemma 8 that \( y_k^\ast = -\frac{\gamma}{\rho_k} \tilde{\lambda}_k \), hence,
\[
\mathcal{H}_k \triangleq \{ h \in \mathcal{W}_k : [y_k^\ast]_h < 0 \} = \{ h \in \mathcal{W}_k : [\tilde{\lambda}_k]_h < 0 \},
\]

since \( -\frac{\gamma}{\rho_k} > 0 \), i.e., the same indices of \( y_k^\ast \) and \( \tilde{\lambda}_k \) will be negative, and these components are given by the set \( \mathcal{H}_k \), which proves 1. Next, \( [y_k^\ast]_h < 0 \) inserted into (19) gives \( \alpha^h_k \in [0, 1), \forall h \in \mathcal{H}_k \), hence, Lemma 9 can be used and gives the same ordering of \( \{ \alpha^h_k \}_{h \in \mathcal{H}_k} \) 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 \( y_k^\ast \), 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 \( M_kM^T_k \) is singular, regaining nonsingularity of \( M_{k+1}M^T_{k+1} \).

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
\[
\begin{align*}
\text{minimize} \quad & \frac{1}{2} \lambda^T M M^T \lambda + d^T(\theta) \lambda, \\
\text{subject to} \quad & \lambda \geq 0
\end{align*}
\]

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

1. \( \nu \leftarrow L^{-1}(f_0 + f) \); \( \nu \leftarrow (b + W\theta + Mv) \)
2. \( k \leftarrow 1, \mathcal{W}_k \leftarrow \emptyset; \lambda_k \leftarrow 0; \)
3. while \( \text{true} \) do
4. if \( M_kM^T_k \) is singular then go to step 16
5. \( \lambda^\ast_k \leftarrow \text{solution to KKT-system (15)} \)
6. if \( \lambda^\ast_k \geq 0 \) then
7. \( \mu_k \leftarrow \mu^1(\lambda^\ast_k | \mathcal{W}_k) + d \)
8. if \( \mu_k \leq -\epsilon \) then
9. return \( \lambda^\ast, x^\ast \leftarrow -L^{-1}(M^T\lambda^\ast_k + v), \mathcal{W}_k \)
10. else \( i \leftarrow \text{arg min}_{i \in \mathcal{W}_k} \{ \mu_i \}; \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \cup \{ i \} \)
11. \( \lambda_{k+1} \leftarrow \lambda^\ast_k \)
12. else \( l \leftarrow \text{arg min}_{h \in \mathcal{W}_k : [l]_h > 0} \{ \frac{|l|}{[l]_h} \}; \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{ l \} \)
13. \( \beta_k \leftarrow -\frac{|l|}{[l]_h}; \lambda_{k+1} \leftarrow \lambda_k + \beta_k(\lambda^\ast_k - \lambda_k) \)
14. \( k \leftarrow k + 1 \)
15. end while
16. \( \tilde{\lambda}_k \leftarrow \text{solution to } M_kM^T_k \mathcal{H}_k = 0, \quad d^T_k \tilde{\lambda}_k \leq 0 \)
17. if \( \tilde{\lambda}_k \geq 0 \) then
18. return primal infeasible
19. else \( l \leftarrow \text{arg min}_{h \in \mathcal{W}_k : [l]_h < 0} \{ [l]_h \}; \mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{ l \} \)
20. \( \tilde{\beta}_k \leftarrow -\frac{|l|}{[l]_h}; \lambda_{k+1} \leftarrow \lambda_k + \tilde{\beta}_k \lambda_k \)
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 \( k(\theta) \) be the number of iterations needed by Algorithm 2 to terminate and let \( [\tilde{\mathcal{W}}_k(\theta)]_{k=1}^{k(\theta)} \) be the corresponding working-set sequence produced by Algorithm 2. Then Algorithm 1 terminates in \( k(\theta) \) iterations and produces the working-set sequence \( [\tilde{\mathcal{W}}_k(\theta)]_{k=1}^{k(\theta)} \), \( \forall \theta \in \Theta_0 \).

**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 \( M_kM^T_k \) 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
Dantzig’s method.

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].

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 $i \neq 1, 2$.

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.