IET Control Theory & Applications

Research Article

IET Journals
The Institution of Engineering and Technology

Fast l_1 model predictive control based on sensitivity analysis strategy

ISSN 1751-8644
Received on 13th May 2019
Revised 3rd October 2019
Accepted on 2nd December 2019
E-First on 16th December 2019
doi: 10.1049/iet-cta.2019.0556
www.ietdl.org

Hamid Kalantari[1] ✉, Mohsen Mojiri[1], Stevan Dubljevic[2], Najmeh Zamani[1]

[1]*Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan 84156-83111, Iran*
[2]*Department of Chemical and Materials Engineering, University of Alberta, Edmonton, Alberta, T6G 2V4, Canada*
✉ *E-mail: hamid.kalantari@ec.iut.ac.ir*

Abstract: This study proposes a new method based on sensitivity analysis to solve a series of sequential parametric linear programmings (LPs) such as that those arise in l_1 model predictive control l_1 (MPC). The main idea is to find a relationship between each of the two successive parametric LPs by using sensitivity analysis strategy. Tolerance analysis-based MPC (TA-l_1 MPC) and sensitivity analysis-based MPC (SA-l_1 MPC) are introduced for reducing computational complexity and runtime. TA-l_1 MPC takes $O(Nn^2)$ operations per step time, where N and n are the prediction horizon and the number of states, respectively. This approach is very faster than generic optimisation methods but it can be applied only for initial conditions that are near to steady-state values. SA-l_1 MPC has not any limitation in usage and it reduces the runtime significantly compared with common solvers. Finally, numerical results indicate the potential of the proposed algorithms.

1 Introduction

Model predictive control (MPC) is a practical method to achieve optimal performance in industrial processes that has been based on online optimisation [1]. To obtain control input $u(k)$ at step time k, a constrained optimisation problem dependent on states and disturbance at step time k is solved for a finite horizon [2, 3]. The objective function is usually expressed as quadratic (l_2 norm) or linear (l_1 norm) form, which leads to quadratic programming (QP) or linear programming (LP), respectively [4]. Quadratic MPC is the oldest and most used type of MPC and has been investigated in many kinds of research [5–9]. MPC based on LP (l_1 MPC) with all its aspects including stability, multiple optima, ideal control, and dead-beat control has been investigated in [4, 10].

l_1 MPC could be used in some problems such as traffic control [11]. But restrictions on the storage memory space and the runtime limit the application of MPC in many real problems such as fast or large scale systems [12]. To date, many suitable algorithms and packages for QP and LP are available which are able to solve l_1 MPC problem. Explicit MPC is a method to compute the optimal controller by using multi-parametric LP [10]. The computational complexity and storage space in the explicit MPC approach grows exponentially with the dimension of the system. Therefore, this method is suitable for small dimension systems [13]. Some studies such as [12, 14, 15] have been tried to overcome the weakness of the explicit method. Also, currently there are various methods to solve LP by using fast online optimisation algorithms such as the fast gradient method [16–18], active set method [19] and interior point method [20]. These methods decrease runtime but the computational complexity of them is $O(N^3n^3)$ and in some special case is $O(Nn^3)$ [13, 21].

Although various methods and effective strategies have been represented to solve the MPC problem fast, the MPC controller is not used in many fast or large scale problems yet. Also, by reducing computational complexity and runtime, the cost of implementation of MPC reduces, too. Therefore, the research to explore new strategies for reducing computational complexity or runtime of the MPC problem is very important, particularly for fast or large scale systems.

In this paper, a new approach based on the sensitivity analysis in the simplex method is introduced to solve l_1 MPC. The sensitivity analysis is used to investigate the robustness of optimal solution of LP with respect to uncertainties in objective function coefficients, the constraint matrix and right-hand side (RHS) parameters of constraints [22]. Also, if variations in coefficients and parameters lead to a variation of the optimal solution, sensitivity analysis can be applied to compute new optimal solution [23]. This capability of sensitivity analysis helps us to represent a new very fast solver for l_1 MPC.

l_1 MPC formulation is converted to a parametric LP so that coefficients of objective function and constraint matrix in the simplex table are constant, but RHS parameters are a linear combination of the state, disturbance signal and reference signal [10, 24]. Therefore, tolerance analysis and sensitivity analysis of RHS parameters can be applied to establish a correlation between the solutions of the consecutive LPs. In other words, the optimal solution at step $k+1$ can be calculated by using RHS tolerance or sensitivity analysis and the optimal solution at step k.

Tolerance analysis-based l_1 MPC, namely TA-l_1 MPC, is developed for the initial conditions close to the steady-state values of the system. It is proved that computational complexity of the proposed method is $O(Nn^2)$. So runtime would be reduced significantly compared with fast online optimisation algorithms mentioned before. Of course, application of TA-l_1 MPC would be limited with initial conditions and therefore it could be used in fine-tuning online optimisation.

Sensitivity analysis-based l_1 MPC, namely SA-l_1 MPC, would be developed to remove the restrictions on the initial conditions. Sensitivity analysis algorithm can be applied to solve a perturbed LP by using the solution of the nominal LP. So, if LP in step $k+1$ considered as a perturbed LP in step k, the solution of LP in step $k+1$ can be calculated by using the solution of LP in step k. The correlation between the two consecutive parametric LPs is dependent on the changes in states, disturbance, and reference signals. Hence, by choosing suitable sampling time (sufficiently small), the correlation between two consecutive parametric LPs would be increased and consequently, the runtime could be reduced compared with fast online optimisation methods significantly.

Sections of this paper are as follows. In Section 2, sensitivity analysis for LP is reviewed and the procedure of conversion l_1 MPC problem to a parametric LP is presented. In Section 3, two algorithms based on tolerance analysis and sensitivity analysis are proposed to reduce the complexity and the runtime of l_1 MPC.

Inputs S^{-1},b^*,Δb
Output optimal solution of perturbed LP (3)
1. Compute (4)
2.

if $b^*_\Delta \geq 0$
    Go to 4
else
    Go to 3

3. Solve new LP (6) by dual simplex method
4. Stop

Fig. 1 *Algorithm 1: Solution of perturbed LP (3) by sensitivity analysis*

Numerical results and analysis of the proposed methods are explained in Section 4 and Section 5 concludes the paper.

2 Problem formulation

2.1 Sensitivity analysis

Sensitivity analysis, whose basic role in LP is well known, is a fundamental tool to investigate the uncertainties in costs, prices, resource availabilities, demands and other elements of LP models [25].

Consider the standard LP

$$\begin{aligned} \max\ z &= c^{\mathrm{T}}y \\ \text{s.t. } My &\leq b \\ y &\geq 0 \end{aligned} \tag{1}$$

in which, $c = [c_1\, c_2 \dots c_n]^{\mathrm{T}} \in R^n$ is the objective function coefficients vector, M is a $m \times n$ full row rank matrix which is named constraint matrix, $b = [b_1\, b_2 \dots b_m]^{\mathrm{T}} \in R^m$ represents the values of RHS terms so that $b_i \geq 0, i = 1, 2, \dots, m$ and $y = [y_1\, y_2 \dots y_n]^{\mathrm{T}} \in R^n$ is decision variable. The vector inequalities are element-wise. Also, let S denote an optimal basis matrix of (1). In other words

$$b^* = S^{-1}b \tag{2}$$

in which b^* is optimal solution of LP (1). S can be obtained by multiplying the elementary matrices used in the simplex procedure and it can be extracted from the simplex table [23].

Sensitivity analysis in LP by the tolerance approach considers simultaneous and independent changes in the objective function coefficients and RHS terms [25]. In particular, for the RHS terms, tolerance analysis finds the largest changes in these terms simultaneously and independently while the optimal basis does not vary.

To address the perturbations of the RHS terms, the tolerance approach focuses on the following perturbed problem:

$$\begin{aligned} \max\ z &= c^{\mathrm{T}}y \\ \text{s.t. } My &\leq b + \Delta b \\ y &\geq 0 \end{aligned} \tag{3}$$

where $\Delta b = [\Delta b_1\, \Delta b_2 \dots \Delta b_m]^{\mathrm{T}} \in R^m$, and $\Delta b_i \ \ i = 1, 2, \dots, m$ is arbitrary amount of perturbation on b_i. Now, consider the interval $[l_i, u_i]$ with the property that the same basis is optimal in (3) as long as the value of each Δb_i belongs to $[l_i, u_i]$. In other words, the same basis is optimal in (3) as long as each Δb_i satisfies the condition $l_i \leq \Delta b_i \leq u_i$. This interval has been determined in [25].

It is well known that $b^* \geq 0$ is the sufficient condition for feasibility of LP in the simplex method. In the result, by attention to (2) and feasibility condition, the optimal solution of perturbed LP (3), denoted by b^*_Δ, can be calculated as

$$b^*_\Delta = b^* + S^{-1}\Delta b \tag{4}$$

so that

$$S^{-1}\Delta b + b^* \geq 0\,. \tag{5}$$

Consequently, allowable changes of each Δb_i will be computed by solving the linear matrix inequality (5).

In addition, if feasibility condition does not establish for perturbed LP (3), i.e. some elements of b^*_Δ be negative, then new LP

$$\begin{aligned} \max\ z &= c^{\mathrm{T}}_\Delta y \\ \text{s.t. } S^{-1}My &\leq b^*_\Delta \\ y &\geq 0 \end{aligned} \tag{6}$$

will be obtained and dual simplex method (see [23]) can be applied for finding the optimal solution of (6). In (6), c_Δ is the objective function coefficients vector in the final table of the simplex method to solve LP (1). In other words, when LP (1) is solved by simplex method, objective function coefficients vector is converted from c in initial LP (1) to c_Δ in final table.

Algorithm 1 (see Fig. 1) describes the main steps of solving perturbed LP by sensitivity analysis.

2.2 l_1 MPC formulation

Consider the discrete-time LTI system

$$x(k+1) = Ax(k) + Bu(k) + Ed(k) \tag{7}$$

in which $x \in R^n$ is state, $u \in R^p$ is input, $d \in R^m$ is disturbance and A,B,E are fixed matrices. Also, let $x_r(k)$ be reference signal to be tracked. So, the tracking error is defined $e(k) = x(k) - x_r(k)$, and we have

$$e(k+1) = Ae(k) + Bu(k) + Ed(k) + Ax_r(k) - x_r(k+1) \tag{8}$$

The MPC problem for (8) with measured disturbance can be defined as the following way:

$$\begin{aligned} \min_{u(k+i|k)}\ & J = \| Pe(k+N|k) \|_1 + \sum_{i=1}^{N-1} \| Qe(k+i|k) \|_1 \\ & \quad + \sum_{i=0}^{N-1} \| Ru(k+i|k) \|_1 \\ \text{s.t.}\ & \\ & e(k|k) = e(k) \\ & e(k+i|k) = Ae(k+i-1|k) + Bu(k+i-1|k) \\ & \qquad + Ed(k+i-1|k) + Ax_r(k+i-1|k) \\ & \qquad - x_r(k+i|k) \\ & x_{\min} - x_r(k+i|k) \leq e(k+i|k) \leq x_{\max} - x_r(k+i|k) \\ & u_{\min} \leq u(k+i-1|k) \leq u_{\max},\ \ i = 1, 2, \dots, N \end{aligned} \tag{9}$$

where $Q = \mathrm{diag}(q)$ and $R = \mathrm{diag}(r)$ are diagonal matrices in which $q^{\mathrm{T}} = [q_1, q_2, \dots, q_n] \geq 0, r^{\mathrm{T}} = [r_1, r_2, \dots, r_p] \geq 0$ are vectors with non-negative elements, P is the terminal cost matrix, $x_{\min}$ and $x_{\max}$ are the lower and upper bounds of states, respectively, $u_{\min}$ and $u_{\max}$ are the lower and upper bounds of inputs, respectively, and N is the prediction horizon. The vector inequalities are element-wise.

In [4, 10], the closed-loop stability of l_1 MPC (9) has been investigated. A sufficient condition to guarantee closed-loop stability for (9) is

$$-\| Pw \|_1 + \| PAw \|_1 + \| Qw \|_1 \leq 0 \tag{10}$$

for all $w \in R^n$ [10]. Therefore the stability of (9) is guaranteed by suitable choice of the terminal cost matrix P. Also, [4] has proposed a terminal cost matrix P to guarantee the closed-loop stability of (9).

Measured disturbance $d(k)$ can be included in the prediction model so that $d(k+i-1)$ is prediction of disturbance at time $k+i-1$ based on the measured value $d(k)$. Usually, $d(k+i-1)$ is a linear function of $d(k)$, for instance $d(k+i-1) \equiv d(k)$ where it is assumed that the disturbance is constant over the prediction horizon [6].

We may transform the MPC problem (9) to a LP by introducing auxiliary variables and the following standard approach. The decision variable y could be defined as $y^{\mathrm{T}} = [\bar{\alpha}_1^{\mathrm{T}} \dots \bar{\alpha}_N^{\mathrm{T}}\, \bar{\beta}_1^{\mathrm{T}} \dots \bar{\beta}_N^{\mathrm{T}}\, \bar{\eta}_0^{\mathrm{T}} \dots \bar{\eta}_{N-1}^{\mathrm{T}}\, \bar{\mu}_0^{\mathrm{T}} \dots \bar{\mu}_{N-1}^{\mathrm{T}}]$ where the non-negative auxiliary variables $\bar{\alpha}_i \in R^n$, $\bar{\beta}_i \in R^n$, $\bar{\eta}_{i-1} \in R^p$, $\bar{\mu}_{i-1} \in R^p$, $for\, i = 1, 2, \dots, N$ are defined such that

$$\begin{aligned}
e(k+i|k) &= \bar{\alpha}_i - \bar{\beta}_i \\
|e(k+i|k)| &= \bar{\alpha}_i + \bar{\beta}_i \\
Pe(k+N|k) &= \bar{\alpha}_N - \bar{\beta}_N \\
|Pe(k+N|k)| &= \bar{\alpha}_N + \bar{\beta}_N \\
u(k+i-1|k) &= \bar{\eta}_{i-1} - \bar{\mu}_{i-1} \\
|u(k+i-1|k)| &= \bar{\eta}_{i-1} + \bar{\mu}_{i-1} \\
u(k+N-1|k) &= \bar{\eta}_{N-1} - \bar{\mu}_{N-1} \\
|u(k+N-1|k)| &= \bar{\eta}_{N-1} + \bar{\mu}_{N-1}
\end{aligned} \tag{11}$$

By defining

$$c = -[\underbrace{q^{\mathrm{T}} \dots q^{\mathrm{T}}}_{2(N-1)}\, \mathbf{1}_n^{\mathrm{T}}\, \mathbf{1}_n^{\mathrm{T}}\, \underbrace{r^{\mathrm{T}} \dots r^{\mathrm{T}}}_{2N}]^{\mathrm{T}}.$$

where $\mathbf{1}_n$ is a column vector in R^n that all its elements are 1, the performance index J in (9) can be expressed as $J = -c^{\mathrm{T}} y$. The constraints in (9) can be expressed in the matrix inequality form

$$My \le b_k$$

with

$$M = \begin{bmatrix} -I_{nN} & I_{nN} & \bar{B} & -\bar{B} \\ I_{nN} & -I_{nN} & -\bar{B} & \bar{B} \\ 0 & 0 & -\bar{B} & \bar{B} \\ 0 & 0 & \bar{B} & -\bar{B} \\ 0 & 0 & -I_{pN} & I_{pN} \\ 0 & 0 & I_{pN} & -I_{pN} \end{bmatrix}$$

where

$$\bar{B} = \bar{P} \begin{bmatrix} B & 0_{n\times p} & \cdots & 0_{n\times p} & 0_{n\times p} \\ AB & B & \cdots & 0_{n\times p} & 0_{n\times p} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{N-2}B & A^{N-3}B & \cdots & B & 0_{n\times p} \\ A^{N-1}B & A^{N-2}B & \cdots & AB & B \end{bmatrix}$$

in which

$$\bar{P} = \begin{bmatrix} I_{(N-1)n} & 0_{(N-1)n\times n} \\ 0_{n\times(N-1)n} & P \end{bmatrix}$$

and

$$\begin{aligned}
b_k &= \Gamma e(k) + \Phi \bar{d}(k) + \Phi_1 \bar{x}_r(k) + \Phi_2 \bar{x}_r(k+1) + L \\
&= \begin{bmatrix} -\bar{A} \\ \bar{A} \\ \bar{A} \\ -\bar{A} \\ 0_{2Np\times n} \end{bmatrix} e(k) + \begin{bmatrix} -\bar{E} \\ \bar{E} \\ \bar{E} \\ -\bar{E} \\ 0_{2Nm\times Nm} \end{bmatrix} \bar{d}(k) \\
&\quad + \begin{bmatrix} -\bar{X} \\ \bar{X} \\ \bar{X} \\ -\bar{X} \\ 0_{2Nn\times Nn} \end{bmatrix} \bar{x}_r(k) + \begin{bmatrix} \bar{Z} \\ -\bar{Z} \\ I - \bar{Z} \\ \bar{Z} - I \\ 0_{2Nn\times Nn} \end{bmatrix} \bar{x}_r(k+1) \\
&\quad + \begin{bmatrix} 0_{2Nn\times 1} \\ -\bar{x}_{\min} \\ \bar{x}_{\max} \\ -\bar{u}_{\min} \\ \bar{u}_{\max} \end{bmatrix}
\end{aligned} \tag{12}$$

in which

$$\bar{d}(k) = \left[d(k|k)^{\mathrm{T}}\; d(k+1|k)^{\mathrm{T}}\; \dots\; d(k+N-1|k)^{\mathrm{T}}\right]^{\mathrm{T}}$$

$$\bar{x}_r(k) = \left[x_r(k|k)^{\mathrm{T}}\; x_r(k+1|k)^{\mathrm{T}}\; \dots\; x_r(k+N-1|k)^{\mathrm{T}}\right]^{\mathrm{T}}$$

$$\bar{A} = \bar{P}\left[A^{\mathrm{T}}\; A^{2\mathrm{T}}\; \dots\; A^{N\mathrm{T}}\right]^{\mathrm{T}}$$

and

$$\bar{E} = \bar{P} \begin{bmatrix} E & 0_{n\times m} & \cdots & 0_{n\times m} & 0_{n\times m} \\ AE & E & \cdots & 0_{n\times m} & 0_{n\times m} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{N-2}E & A^{N-3}E & \cdots & E & 0_{n\times m} \\ A^{N-1}E & A^{N-2}E & \cdots & AE & E \end{bmatrix}$$

$$\bar{X} = \bar{P} \begin{bmatrix} A & 0_{n\times n} & \cdots & 0_{n\times n} & 0_{n\times n} \\ A^2 & A & \cdots & 0_{n\times n} & 0_{n\times n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{N-1} & A^{N-2} & \dots & A & 0_{n\times n} \\ A^N & A^{N-1} & \cdots & A^2 & A \end{bmatrix}$$

$$\bar{Z} = \bar{P} \begin{bmatrix} I & 0_{n\times n} & \cdots & 0_{n\times n} & 0_{n\times n} \\ A & I & \cdots & 0_{n\times n} & 0_{n\times n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{N-2} & A^{N-3} & \dots & I & 0_{n\times n} \\ A^{N-1} & A^{N-2} & \cdots & A & I \end{bmatrix}$$

Also, $\bar{x}_{\max} = 1_N \otimes x_{\max}$, $\bar{x}_{\min} = 1_N \otimes x_{\min}$, $\bar{u}_{\max} = 1_N \otimes u_{\max}$ and $\bar{u}_{\min} = 1_N \otimes u_{\min}$. 1_N is a column vector in R^N with all entries being one and $\otimes$ is Kronecker product.

Therefore, the l_1 MPC formulation (9) is converted to the LP

$$\begin{aligned}
\max_{y} \quad & c^{\mathrm{T}} y \\
\text{s.t.} \quad & My \le b_k \\
& y \ge 0
\end{aligned} \tag{13}$$

for $k = 0, 1, 2, \dots$. Note that the minimisation problem (9) has been converted to a maximisation problem by multiplying -1 in objective function coefficients. In (13), M and c are constants and

b_k is a linear combination of error $e(k)$ at step time k and disturbance and reference signals at the prediction horizon $[k, k+1, \ldots, k+N]$. In other words, l_1 MPC problem at each step time leads to a parametric LP. Therefore, by finding a relationship between b_{k+1} and b_k, we can find a relationship between solutions of the parametric LPs at two step times $k+1$ and k. Consequently, we can obtain an optimisation algorithm with less computational complexity than existing approaches.

3 Proposed fast l_1 MPC

In this section, we propose two algorithms to solve l_1 MPC problem by tolerance and sensitivity analysis approaches (Algorithm 1 (Fig. 1)) for reducing computational complexity and runtime.

In the first algorithm, we have used RHS tolerance analysis to reduce the computational complexity for small variations in tracking error e, disturbance signal d and reference signal x_r around initial values at step time $k = 0$. For this method, it is proved that computational complexity to solve l_1 MPC is $O(Nn^2)$.

In the second approach, we represent an algorithm to solve l_1 MPC by using sensitivity analysis. In this method, there is no limitation on signals and initial conditions and the runtime is reduced significantly compared with interior-point, active set, and simplex methods.

3.1 Tolerance analysis-based l_1 MPC

Tolerance analysis is a powerful method in the simplex algorithm for finding permissible variations in RHS so that the optimal basis does not change [22]. Considering (13) and (12), coefficients vector c and constraint matrix M are constant and RHS vector b_k is a linear combination of e, d and x_r. Therefore, due to the tolerance analysis, there exist variations in e, d and x_r which optimal basis matrix S do not change.

Notation: $(Y)_j$ denotes jth row of matrix Y. $e_0(k) \equiv e(0)$, $\bar{d}_0(k) \equiv \bar{d}(0)$, $\bar{x}_{r0}(k) \equiv \bar{x}_r(0)$ and $\bar{x}_{r1}(k) \equiv \bar{x}_r(1)$ for $k \geq 0$ are constant vector sequences.

Definition 1: Infinity norm of the discrete-time signal $x(k)$ for $k = 0, 1, 2, \ldots$ denoted by $\| \cdot \|_\infty$ is defined as

$$\| x \|_\infty = \sup_k \| x(k) \|$$

where $\| \cdot \|$ is Euclidean norm.

Proposition 1: Consider l_1 MPC (9) which is formulated as the parametric LP (13). Let S is the optimal basis matrix and b_0^* is the optimal solution of LP (13) at the initial time $k = 0$. If

$$\begin{aligned}(b_0^*)_j \geq & \| e - e_0 \|_\infty \ \| (S^{-1}\Gamma)_j \| \\ & + \| \bar{d} - \bar{d}_0 \|_\infty \ \| (S^{-1}\Phi)_j \| \\ & + \| \bar{x}_r - \bar{x}_{r0} \|_\infty \ \| (S^{-1}\Phi_1)_j \| \\ & + \| \bar{x}_r - \bar{x}_{r1} \|_\infty \ \| (S^{-1}\Phi_2)_j \|\end{aligned} \tag{14}$$

for $j = 1, 2, \ldots, 2N(2n+p)$, then optimal basis for all the times $k = 0, 1, 2, \ldots$ is same. Moreover, optimal solution of LP (13) at each step time k is computable with $O(N^2n^2)$.

Proof: Using tolerance analysis and (5), there exists Δb so that

$$S^{-1}\Delta b + b_0^* \geq 0. \tag{15}$$

Therefore, we must derive conditions on

$$\begin{aligned}\Delta b_k &= b_k - b_{k-1} \\ &= \Gamma(e(k) - e(k-1)) \\ &\quad + \Phi\big(\bar{d}(k) - \bar{d}(k-1)\big) \\ &\quad + \Phi_1(\bar{x}_r(k) - \bar{x}_r(k-1)) \\ &\quad + \Phi_2(\bar{x}_r(k+1) - \bar{x}_r(k)) \\ \text{for } k &= 1, 2, 3, \ldots,\end{aligned} \tag{16}$$

such that (15) is satisfied. By assuming feasibility condition and same basis for (13) at each step time k and using (4) and (16), we have

$$\begin{aligned}b_k^* &= S^{-1}(\Gamma(e(k) - e(0)) \\ &\quad + \Phi(\bar{d}(k) - \bar{d}(0)) \\ &\quad + \Phi_1(\bar{x}_r(k) - \bar{x}_r(0)) \\ &\quad + \Phi_2(\bar{x}_r(k+1) - \bar{x}_r(1))) + b_0^*\end{aligned} \tag{17}$$

Therefore, to establish a feasibility condition, the inequality

$$\begin{aligned}b_{k,j}^* &= (S^{-1}\Gamma)_j(e(k) - e(0)) \\ &\quad + (S^{-1}\Phi)_j(\bar{d}(k) - \bar{d}(0)) \\ &\quad + (S^{-1}\Phi_1)_j(\bar{x}_r(k) - \bar{x}_r(0)) \\ &\quad + (S^{-1}\Phi_2)_j(\bar{x}_r(k+1) - \bar{x}_r(1)) + b_{0,j}^* \geq 0 \\ \text{for } j &= 1, 2, \ldots, 2N(2n+p) \\ k &= 1, 2, 3, \ldots\end{aligned} \tag{18}$$

must be satisfied. Using the Cauchy–Schwarz inequality, sufficient condition to achieve the feasibility condition is as follows:

$$\begin{aligned}b_{0,j}^* &- \| (S^{-1}\Gamma)_j \| \| e(k) - e(0) \| \\ &- \| (S^{-1}\Phi)_j \| \| \bar{d}(k) - \bar{d}(0) \| \\ &- \| (S^{-1}\Phi_1)_j \| \| \bar{x}_r(k) - \bar{x}_r(0) \| \\ &- \| (S^{-1}\Phi_2)_j \| \| \bar{x}_r(k+1) - \bar{x}_r(1) \| \geq 0 \\ \text{for } j &= 1, 2, \ldots, 2N(2n+p) \\ k &= 1, 2, 3, \ldots\end{aligned} \tag{19}$$

Now, for worst case if

$$\begin{aligned}b_{0,j}^* - & \sup_k (\| (S^{-1}\Gamma)_j \| \| e(k) - e(0) \| \\ & - \| (S^{-1}\Phi)_j \| \| \bar{d}(k) - \bar{d}(0) \| \\ & - \| (S^{-1}\Phi_1)_j \| \| \bar{x}_r(k) - \bar{x}_r(0) \| \\ & - \| (S^{-1}\Phi_2)_j \| \| \bar{x}_r(k+1) - \bar{x}_r(1) \|) \geq 0 \\ \text{for } j & = 1, 2, \ldots, 2N(2n+p)\end{aligned}$$

that is the equivalent to (14), then feasibility condition is established and as a result, optimal basis S will be same at all step times.

Also, by (17) optimal solution can be computable at each step time k. In (17), S^{-1} is $2N(2n+p) \times 2N(2n+p)$, Γ is $2N(2n+p) \times n$, Φ is $2N(2n+p) \times mN$, Φ_1 is $2N(2n+p) \times nN$ and Φ_2 is $2N(2n+p) \times nN$. Consequently computational order can be derived at each step time as $2N(2n^2+np) + 2N^2m(2n+p) + 4N^2(2n^2+np)$ flops. Assuming $n \geq p, m$ computational complexity can be written as $O(N^2n^2)$. □

Using Proposition 1, we propose Algorithm 2 to solve l_1 MPC problem (9) by tolerance analysis as follows:

Algorithm 2 (Solution of l_1 MPC (9) by tolerance analysis):
Offline mode for $k = 0$

Inputs $x(0)$, $\bar{d}(0)$, $\bar{x}_r(0)$

Output S^{-1}, b_0^*, $u(0)$

1. Solve (12) by simplex method

2. Extract S^{-1}, b_0^*, $u(0)$ from simplex table
Online mode for $k = 1, 2, \ldots$
Inputs S^{-1}, b_0^*, $x(0)$, $\bar{d}(0)$, $\bar{x}_r(0)$, $x(k)$, $\bar{d}(k)$, $\bar{x}_r(k)$
Output $u(k|k)$
1. Compute $e(k) - e(0)$, $\bar{d}(k) - \bar{d}(0)$, $\bar{x}_r(k) - \bar{x}_r(0)$
2. Compute b_k^* by (17)
3. Compute $u(k|k) = \bar{\eta}_0 - \bar{\mu}_0$

The features of the proposed TA-l_1 MPC algorithm are as follows:

1. *Complexity*: According to Algorithm 2 and proof of Proposition 1, the computational complexity to solve MPC problem (9) at each step time $k = 1, 2, \ldots$ can be earned as $O(pn + pmN + 2pnN)$ that can be written as $O(Nn^2)$. Because by (11) we need to obtain $\bar{\eta}_0$ and $\bar{\mu}_0$ for computing $u(k|k)$ and by (18) for computing $\bar{\eta}_0$ and $\bar{\mu}_0$, we use only the p number of the elements of b_k^*. Therefore, complexity grows linearly with N to solve l_1 MPC.
2. *Parallel computing*: Because S, Γ, Φ, Φ_1 and Φ_2 are constant matrices, we could obtain minimum usage communication links for solving MPC as centralised. The required information, such as $x(k|k)$, $\bar{d}(k)$ and $\bar{x}_r$, for computing $u(k|k)$ would be determined by (17). Therefore, LP (13) can be divided to smaller dimension problems and be solved by parallel computing. As a result, large scale systems could be controlled with distributed processors with performance such as centralised approach. In this approach by (11), $\bar{\eta}_{0,s}$ and $\bar{\mu}_{0,s}$ which are the sth elements of $\bar{\eta}_0$ and $\bar{\mu}_0$, respectively, are used for computing each element of $u(k|k)$, that be represented as $u_s(k|k)$. On the other side, only one of $\bar{\eta}_{0,s}$ or $\bar{\mu}_{0,s}$ is in the basis and another has zero value. So each $u_s(k|k)$ for $s = 1, 2, \ldots, p$ can be computed with $O(n + mN + 2nN)$ which can be written as $O(Nn)$. Consequently, p numbers of distributed processors can control large scale system with $O(Nn)$ flops for each processor.
3. Proposition 1 and Algorithm 2 introduce an approach for solving MPC locally. In other words, this approach is valuable for initial conditions close to steady-state values. Also, the variations of the steady-state values must be sufficiently small. If variations of e, d and x_r respect to their values at initial time $k = 0$ be sufficiently small such that (14) satisfied, then computational order at each step time will be $O(Nn^2)$.
4. In this approach the optimal basis and the optimal basis matrix, S, should be same for all the times. On the other side, in the stability problem where d is zero and error vector e vanishes to zero, the optimal basis cannot be the same for all the times. Because in this case initial error $e(0)$ begin from non-zero values and accordingly they are in the basic variables. But all the elements of the error vector e exit from basis when they are vanishing to zero in infinity time. Therefore, the optimal basis and the optimal basis matrix S change at some step times. So this approach does not use for tracking problem. But this approach is useful for online optimisation problems with disturbance signals such as traffic control.

3.2 Sensitivity analysis-based l_1 MPC

As mentioned, TA-l_1 MPC could not be used for some applications and its application is restricted for some initial conditions and disturbance signals by attention to (14). So we pursue to find an algorithm without these limitations. Sensitivity analysis can be applied to achieve this goal. In TA-l_1 MPC, it is proven that complexity is $O(Nn^2)$, but in this approach reducing of the runtime is illustrated by numerical simulation and is compared with common solvers for l_1 MPC. Consider LP (13) and let S_k be optimal basis matrix for $k = 0, 1, 2, \ldots$. For $\Delta b_{k+1} = b_{k+1} - b_k$, b_{k+1}^{Δ} can be computed by (4) as follows:

$$b_{k+1}^{\Delta} = b_k^* + S_k^{-1}\Delta b_{k+1} \tag{20}$$

If feasibility condition is satisfied, i.e. $b_{k+1}^{\Delta} \geq 0$, the optimal solution b_{k+1}^* is equal to b_{k+1}^{Δ} and therefore SA-l_1 MPC is converted to TA-l_1 MPC and its complexity will be $O(Nn^2)$. Otherwise, i.e. some elements of b_{k+1}^{Δ} is negative, new LP (21) must be solved by dual simplex method.

$$\begin{aligned} \max_{y} \quad & c_k^{\mathrm{T}} y \\ \text{s.t.} \quad & S_k^{-1} M_k y \leq b_{k+1}^{\Delta} \\ & y \geq 0 \end{aligned} \tag{21}$$

In (21) $M_k = S_{k-1}^{-1} M_{k-1}$ for $k = 1, 2, \ldots$ and $M_0 = S_0^{-1} M$. Also c_k is objective function coefficients in final table in step k. To solve (21) by dual simplex method, some elementary row operations are applied on S_k^{-1} that can be represented with elementary matrices $E_{k1}, E_{k2}, \ldots, E_{kl}$ and as a result

$$S_{k+1}^{-1} = E_{kl} \ldots E_{k2} E_{k1} S_k^{-1} \tag{22}$$

where l is the number of iterations to solve dual simplex. In (22) $E_{ki} = I - w_{ki} e_v^{\mathrm{T}}$ for $i = 1, 2, \ldots, l$ where $w_{ki} \in R^{2N(2n+p)}$ is constructed in the final simplex table at step time k and $e_v \in R^{2N(2n+p)}$ is the standard basis vector which all of its elements are zero except vth element that is '1'. So, optimal solution at step time $k + 1$ can be obtained by

$$b_{k+1}^* = E_{kl} \ldots E_{k2} E_{k1} b_{k+1}^{\Delta} \tag{23}$$

The above derivation is summarised in Algorithm 3 (see Fig. 2).

By attention to the special form of E_{ki}, complexity of calculation of $E_{k1} S_k^{-1}$ is $O(N^2(2n + p)^2)$. Thus complexity of computing (22) is $O(N^2(2n + p)^2 l)$. Therefore, the complexity of step 4 in Algorithm 3 (Fig. 2) is $O(N^2(2n + p)^2 l)$, too. And as a result, by assuming $n \geq p, m$ complexity of Algorithm 3 (Fig. 2) is $O(N^2 n^2 l)$. The number of iterations, l, is dependent to quantity of closeness b_{k+1}^{Δ} to b_{k+1}^* that is related to quantity of Δb_{k+1}.

Offline mode for $k = 0$
Inputs $x(0), \bar{d}(0), \bar{x}_r(0)$
Output $S_0^{-1}, b_0{}^*, u(0)$
1. Solve (13) by simplex method
2. Extract $S_0^{-1}, b_0{}^*, u(0)$ from simplex table
Online mode for $k = 1, 2, \ldots$
Inputs S_0^{-1}, $b_0{}^*$, $x(k-1)$, $\bar{d}(k-1)$, $\bar{x}_r(k-1)$, $x(k)$, $\bar{d}(k)$, $\bar{x}_r(k)$
Output $u(k|k)$
1. Compute

$$\begin{aligned} \Delta e(k) &= e(k) - e(k-1) \\ \Delta \bar{d}(k) &= \bar{d}(k) - \bar{d}(k-1) \\ \Delta \bar{x}_r(k) &= \bar{x}_r(k) - \bar{x}_r(k-1) \end{aligned}$$

2. Compute Δb_k by (16)
3. Compute b_k^{Δ} by (20)
4.

if $b_k^{\Delta} \geq 0$
  $b_k^* = b_k^{\Delta}$
  $S_k^{-1} = S_{k-1}^{-1}$
else
  Solve (21) by dual simplex
  Extract S_k^{-1} and b_k^*

5. Compute $u(k|k) = \bar{\eta}_0 - \bar{\mu}_0$

Fig. 2 *Algorithm 3: Solution of l_1 MPC (9) by sensitivity analysis*

Table 1 Comparison of runtime

Method	Max runtime, ms	Mean runtime, ms
active set	212	160
interior-point	105	78
simplex	125	90
TA-l_1 MPC	0.3	0.1

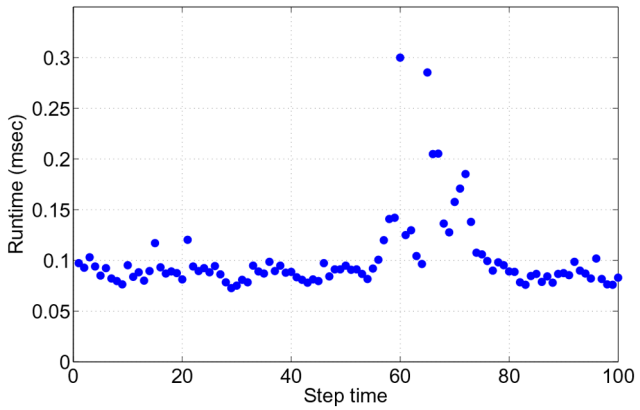

Fig. 3 *Runtime of TA-l_1 MPC with $N = 5$ and $T_s = 5$ s*

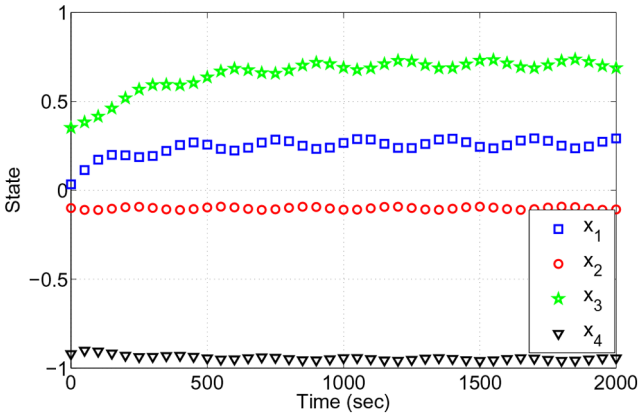

Fig. 4 *States for* $x_0 = [0.03 \quad -0.1 \quad 0.35 \quad -0.9]^{\mathrm{T}}$, $d(k) = 0.1 + 0.02\sin(0.1k)$ *and* $T_s = 5$ *s*

Optimal basis matrix S_k has all information about solved LP (13) at step time k. Therefore, if the variations of e, d and x_r are small between two consecutive step times k and $k+1$, b^{Δ}_{k+1} will be very close to optimal solution b^{*}_{k+1}. In other words, smallness of $\Delta e(k)$, $\Delta\bar{d}(k)$ and $\Delta\bar{x}_r(k)$ causes smallness of Δb_k and so the number of iterations to solve (21) by the dual simplex would be small. And smallness of the number of iterations causes to reduce the runtime of the algorithm.

By choosing suitable sampling time, $\Delta e(k)$, $\Delta\bar{d}(k)$, and $\Delta\bar{x}_r(k)$ would be small and thus the number of iterations and runtime can be reduced significantly. Therefore, choosing the sampling time is an important issue to increase the performance of sensitivity analysis to solve l_1 MPC. Of course, the suitable sampling time is dependent on the behaviour of system and type of discretisation of the continuous-time system. This problem will be investigated in numerical results.

3.3 Offline mode considerations

In the offline mode of proposed algorithms, one LP dependent on initial conditions is solved by the simplex method. Therefore, disturbance and state vectors at $k = 0$ must be available that can be measured or estimated before the starting of the algorithms. The computational complexity of the simplex method is $O(2^{v})$, in which v is the size of the problem, but in practice, it is linearly proportional to v ($O(v)$) (see pp. 163–169 of [23]).

If $x(0)$ and $\bar{d}(0)$ are not accessible before the starting of the proposed algorithms, those would be measured at the beginning of the operation of the system and the simplex method will be solved. Therefore, if it is impossible to solve offline the simplex problem at $k = 0$, online solving would be led to a constant delay (equal to runtime) in computations. The amount of destruction of this delay is related to the dynamics of the system and sampling time. If the system is very fast and the sampling time is very small maybe, this delay reduces the performance and could even be caused instability. But it does not usually reduce the performance of the algorithms significantly. Because the runtime of the simplex and so the delay is usually small compared with sampling time. Thus it seems this situation must be investigated in each problem individually.

4 Numerical results

In this section, two examples are represented to investigate and to indicate properties of proposed methods for fast l_1 MPC. Also, we compare proposed methods with interior-point, active set, and simplex methods. The computations are done by an Intel (R) Core(TM) i5-4200M CPU 2.5 GHz processor and MATLAB software has been used by '*linprog*' command for interior-point, active set and simplex methods.

Example 1: Consider LTI discrete-time system

$$x(k+1) = Ax(k) + Bu(k) + Ed(k)$$

where

$$A = \begin{bmatrix} 0.97 & 0 & 0.02 & 0 \\ 0 & 0.96 & 0 & 0.02 \\ 0 & 0.03 & 0.97 & 0 \\ 0.01 & 0.04 & 0 & 0.98 \end{bmatrix}, \quad B = \begin{bmatrix} 24.6 & 0.5 \\ -0.2 & 32.8 \\ 0.5 & 49.5 \\ -19.8 & -0.5 \end{bmatrix}$$

$$E = [0.1 \quad 0.1 \quad 0.1 \quad -0.1]^{\mathrm{T}}$$

and $d(k) = 0.1 + 0.02\sin(0.1k)$. This is obtained from a continuous-time LTI system by zero-order hold (ZOH) method and sampling time $T_s = 5$ s. Lower and upper bounds for states and control inputs are as follows:

$$u_{\min} = [-0.001 \quad -0.001]^{\mathrm{T}}, \; u_{\max} = [0.001 \quad 0.001]^{\mathrm{T}}$$

$$x_{\min} = [-1 \quad -1 \quad -1 \quad -1]^{\mathrm{T}}$$

$$x_{\max} = [1 \quad 1 \quad 1 \quad 1]^{\mathrm{T}}$$

Also, let $x_r(k) = 0$ and $Q = \mathrm{diag}([0.5 \;\; 0.1 \;\; 0.05 \;\; 0.2]^{\mathrm{T}})$, $R = 0.1I_2$ and $P = 20\,Q$. For $x_0 = [0.03 \quad -0.1 \quad 0.35 \quad -0.9]^{\mathrm{T}}$ that satisfies (14) and $N = 5$, the runtime of TA-l_1 MPC method is compared with the active set, interior-point and simplex methods. The runtime of the Algorithm 2 in offline mode is 80 ms and runtime in online mode has been illustrated in Table 1.

Because the complexity of TA-l_1 MPC method is $O(Nn^2)$, and the complexity of active set and interior-point is $O(N^3n^3)$ and for the simplex method is $O(2^{Nn})$, the runtime of TA-l_1 MPC is very less than other methods. Fig. 3 illustrates runtime of tolerance algorithm for all step times. Runtime is about 0.1 ms in most of the step times. Of course, as seen in Fig. 4, the initial state is close to the steady state for the fulfilment of (14).

For initial state $x_0 = [0 \;\; 0 \;\; 0 \;\; 0]^{\mathrm{T}}$, the runtime is indicated by using SA-l_1 MPC method in Fig. 5. In this case, since the initial state is not close to steady state and therefore (14) cannot be satisfied, TA-l_1 MPC method cannot be applied. The runtime of the offline mode is 75 ms that is very small compared with sampling time and so cannot reduced the performance of the Algorithm 3 (Fig. 2). By attention to Fig. 5, the runtime is between 4 and 10 ms for initial step times from zero until 250 s. In the other side, the

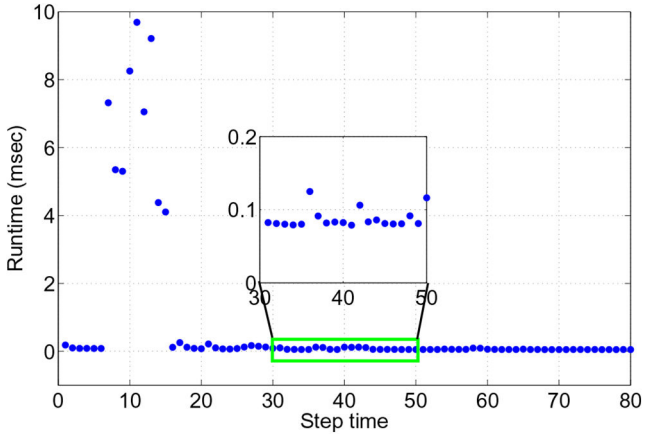

Fig. 5 *Runtime of SA-l_1 MPC for $N = 5$ and $T_s = 5$ s*

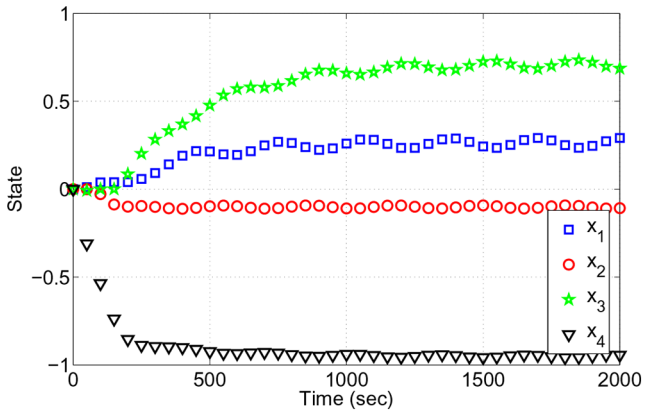

Fig. 6 *States for $x_0 = [0\ 0\ 0\ 0]^T$, $d(k) = 0.1 + 0.02\sin(0.1k)$ and $T_s = 5$ s*

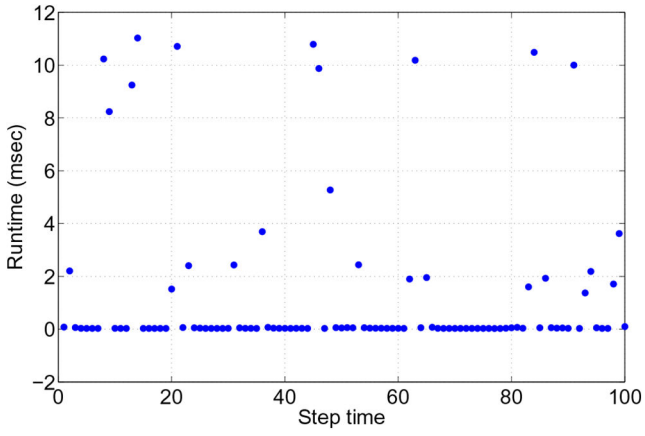

Fig. 7 *Runtime of SA-l_1 MPC for $d(k) = 0.1 + 0.2\sin(0.1k)$, $N = 5$ and $T_s = 5$ s*

behaviour of the system is in the transient mode in interval time $[0, 250]$ (see Fig. 5). According to Figs. 5 and 6, at the end of the transient mode and by approaching the states of the system to steady-state, from 250 to 2000 s, runtime decreases to less than 0.3 ms. In other words, in this example by approaching behaviour of the system to the steady- state, SA-l_1 MPC method with the complexity of $O(N^2n^2\mathrm{l})$ is converted to TA-l_1 MPC method with the complexity of $O(Nn^2)$ and this is the reason of reducing the runtime. But this result is not valid for all problems. For example if $d(k) = 0.1 + 0.2\sin(0.1k)$, runtime obtains as Fig. 7. In this case, due to large variations of d, (14) cannot be satisfied and so the number of iterations to solve dual simplex does not decrease in steady state.

Table 2 and Fig. 8 show comparison between SA-l_1 MPC and interior-point methods to solve above problem for $x_0 = [0\ 0\ 0\ 0]^T$.

Table 2 Comparison of runtime between SA-l_1 MPC method and interior-point method

Method	Max runtime, ms	Mean runtime, ms
interior-point	102	75
SA-l_1 MPC	9.7	0.1

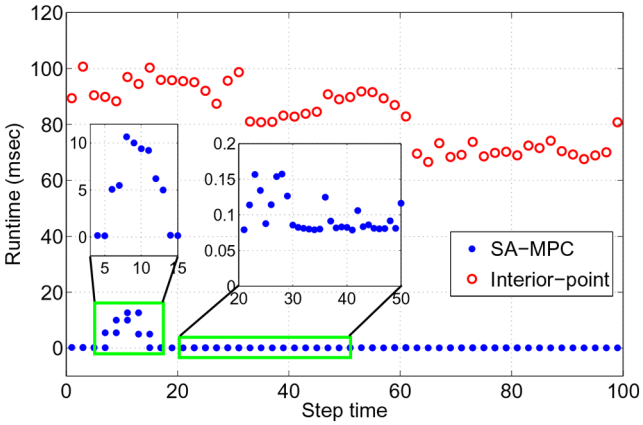

Fig. 8 *Comparison of runtime between SA-l_1 MPC and interior-point methods for $N = 5$ and $T_s = 5$ s*

Table 3 Effect of prediction horizon (N) on runtime in SA-l_1 MPC for $T_s = 5$ s

N	Max runtime, ms	Mean runtime, ms
5	10	0.1
10	28	0.8
15	49	1.7
20	75	3
25	105	4.1
30	255	7.5

Table 4 Effect of sampling time (T_s) on runtime in SA-l_1 MPC for $N = 10$

T_s, s	Max runtime, ms	Mean runtime, ms	$l_{\max}$
0.1	3	0.1	1
0.5	14	0.3	3
1	19	0.4	4
2.5	24	0.45	4
5	28	0.8	5
10	46	1.7	8
20	40	4.1	8
30	43	7.1	8
50	61	9.5	11
70	59	10	11
100	64	14	11

The runtime of SA-l_1 MPC method reduces in steady state but the runtime of the interior-point method is almost fixed in all step times. This advantage in SA-l_1 MPC method stems from sensitivity strategy to solve consecutive LPs and its reason is the reducing the number of iteration l and as a result, the conversion of the complexity from $O(N^2n^2\mathrm{l})$ to $O(Nn^2)$. For $d(k) = 0$, $x_r(k) = 0$ and $x_0 = [-1\ \ 1\ \ -1\ \ 1]^T$, the effect of sampling time and prediction horizon (N) on runtime in SA-l_1 MPC method has been investigated. As seen in Table 3, it seems that maximum and mean runtime is proportional to N^2 approximately.

Also, Table 4 shows the effect of sampling time on runtime. By choosing a small sampling time, the runtime is reduced significantly. Mean runtime is proportional to sampling time as linearly, but two jumps occur in maximum runtime by changing

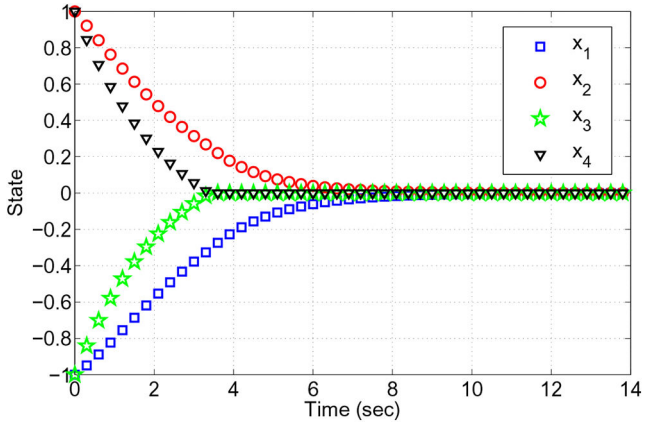

Fig. 9 *States of the four-tank system for* $x_0 = [-1\ 1\ -1\ 1]^T$, $N = 10$ *and* $T_s = 100$ *ms*

Table 5 Comparison of runtime of the various methods for four-tank plant for $N = 10$ and $T_s = 100$ ms

Method	Max runtime, ms	Mean runtime, ms
active set	200	170
interior-point	73	50
simplex	70	40
SA-l_1 MPC	7	0.3

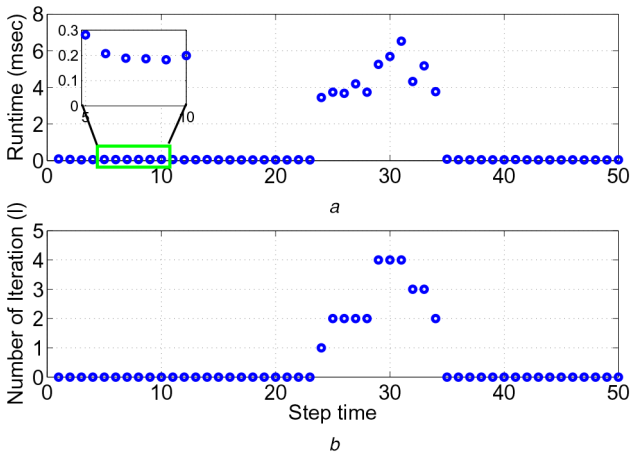

Fig. 10 *Illustration of characteristics of SA-l1MPC for* $N = 10$ *and* $T_s = 100$ *ms*

(a) Runtime, ***(b)*** The number of iteration

sampling time from 5 to 10 s and from 30 to 50 s. Maximum runtime can be divided into three segments that is constant in each segment approximately. For example, sampling time between 1and 5 s is suitable to achievement runtime about 25 ms or between 10 and 30 s for about 45 ms runtime. Also, the maximum runtime is proportional to the maximum number of iteration (l_{max}).

Example 2: Consider the continuous-time model of the four-tank plant that has been investigated in [16]. Using the ZOH method with a sampling period of 100 ms, the discrete-time model is obtained in the following:

$$x(k+1) = Ax(k) + Bu(k)$$

where

$$A = \begin{bmatrix} 0.94 & 0 & 0.04 & 0 \\ 0 & 0.93 & 0 & 0.04 \\ 0 & 0 & 0.96 & 0 \\ 0 & 0 & 0 & 0.96 \end{bmatrix}, \quad B = \begin{bmatrix} 0.32 & 0.03 \\ 0.03 & 0.4 \\ 0 & 1.22 \\ 1.31 & 0 \end{bmatrix}$$

Lower and upper bounds for states and control inputs are as follows:

$$u_{min} = [-0.01 \quad -0.01]^T,\ u_{max} = [0.01 \quad 0.01]^T$$

$$x_{min} = [-1 \quad -1 \quad -1 \quad -1]^T$$

$$x_{max} = [1 \quad 1 \quad 1 \quad 1]^T$$

and $Q = 0.5\,I_4$, $R = 0.1\,I_2$ and $P = 25\,I_4$. By using SA-l_1 MPC, the runtime of the offline mode is 56 ms and the maximum runtime and mean runtime in online mode are 7 and 0.3 ms, respectively, for $N = 10$ and $x_0 = [-1\ 1\ -1\ 1]^T$. Also, by attention to Fig. 9 and the time constant of this system, $T_s = 100$ ms is suitable. As a result, by using SA-l_1 MPC, the performance of the control does not decrease significantly in practice. However, Table 5 shows that the runtime of the interior-point method is about 50% of sampling time and for the simplex method it is around 40%. Thus, these methods decrease the performance of the MPC significantly. Also, the active-set method has the runtime around 170 ms that is greater than the sampling time and so this method can not be used for this system.

Fig. 10 illustrates the runtime and the number of iteration. As seen in Section 3.2, the computational complexity of SA-l_1 MPC for $l > 0$ is $O(N^2n^2l)$, and so the runtime is proportional to the number of iteration (l) for $l > 0$. Also, for $l = 0$, SA-l_1 MPC is converted to TA-l_1 MPC with $O(Nn^2)$. As seen in Fig. 10, because the number of iterations are small so the runtime is very less than other methods and at each step time that $l = 0$ the runtime is very small even in comparison with $l > 0$.

5 Conclusions

In this study, two new algorithms based on sensitivity analysis are proposed to solve l_1 MPC. The complexity of TA-l_1 MPC method is $O(Nn^2)$ and so this algorithm is very fast in comparison with common methods. Also, this approach would be faster by using parallel computing that had been explained in Section 3. Despite of the limitations of TA-l_1 MPC approach, this approach is suitable for fine tuning of large scale systems such as urban traffic control. SA-l_1 MPC method has not any limitations and can be applied for a wide range of problems but is slower than TA-l_1 MPC method. Of course, it is very faster from common solvers such as interior-point and active set methods.

6 References

[1] Mayne, D.: 'Model predictive control: recent developments and future promise', *Automatica*, 2014, **50**, (12), pp. 2967–2986

[2] Camacho, E., Alba, C.: '*Model predictive control*' (Springer Science & Business Media, Berlin, 2007, 2nd edn.)

[3] Maciejowski, J.: '*Predictive control: with constraints*' (Prentice-Hall, Harlow, England, 2002, 1st edn.)

[4] Rao, C., Rawlings, J.: 'Linear programming and model predictive control', *J. Process Control*, 2000, **10**, (2), pp. 283–289

[5] Scokaert, P., Rawlings, J.: 'Constrained linear quadratic regulation', *IEEE Trans. Autom. Control*, 1998, **43**, (8), pp. 1163–1169

[6] Bemporad, A., Morari, M., Dua, V.: 'The explicit linear quadratic regulator for constrained systems', *Automatica*, 2002, **38**, (1), pp. 3–20

[7] Lopes, R., Mendes, E., Torres, L.: 'Constrained robust model predicted control of discrete-time Markov jump linear systems', *IET Control Theory Applic.*, 2019, **13**, (4), pp. 517–525

[8] Xu, J., Boom, T., Schutter, B.: 'Model predictive control for stochastic max-plus linear systems with chance constraints', *IEEE Trans. Autom. Control*, 2018, **64**, (1), pp. 337–342

[9] Cavanini, L., Cimini, G., Ippoliti, G.: 'Computationally efficient model predictive control for a class of linear parameter varying systems', *IET Control Theory Applic.*, 2018, **12**, (10), pp. 1384–1392

[10] Bemporad, A., Borrelli, F., Morari, M.: 'Model predictive control based on linear programming – the explicit solution', *IEEE Trans. Autom. Control*, 2002, **47**, (12), pp. 1974–1985

[11] Zhou, Z., Schutter, B., Lin, S.: 'Two-level hierarchical model-based predictive control for large-scale urban traffic networks', *IEEE Trans. Control Syst. Technol.*, 2017, **25**, (2), pp. 496–508

[12] Zeilinger, M., Jones, C., Morari, M.: 'Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization', *IEEE Trans. Autom. Control*, 2011, **56**, (7), pp. 1524–1534

[13] Wang, Y., Boyd, S.: 'Fast model predictive control using online optimization', *IEEE Trans. Autom. Control*, 2010, **18**, (2), pp. 267–278
[14] Summers, S., Jones, C., Lygeros, J.: 'A multiresolution approximation method for fast explicit model predictive control', *IEEE Trans. Autom. Control*, 2011, **59**, (11), pp. 2530–2541
[15] Bemporad, A.: 'A multiparametric quadratic programming algorithm with polyhedral computations based on nonnegative least squares', *IEEE Trans. Autom. Control*, 2015, **60**, (11), pp. 2892–2903
[16] Zhou, X., Li, C., Huang, T.: 'Fast gradient-based distributed optimisation approach for model predictive control and application in four-tank benchmark', *IET Control Theory Applic.*, 2015, **9**, (10), pp. 1579–1586
[17] Richter, S., Jones, C., Morari, M.: 'Computational complexity certification for real-time MPC with input constraints based on the fast gradient method', *IEEE Trans. Autom. Control*, 2012, **57**, (6), pp. 1391–1402
[18] Patrinos, P., Bemporad, A.: 'An accelerated dual gradient-projection algorithm for embedded linear model predictive control', *IEEE Trans. Autom. Control*, 2014, **59**, (1), pp. 18–33
[19] Ferreau, H., Bock, H., Diehl, M.: 'An online active set strategy to overcome the limitations of explicit MPC', *Int. J. Robust Nonlinear Control*, 2008, **18**, (8), pp. 816–830
[20] Domahidi, A., Zgraggen, A., Zeilinger, M.: 'Efficient interior point methods for multistage problems arising in receding horizon control'. 51st IEEE Conf. on Decision and Control, Hawaii, USA, 2012, pp. 668–674
[21] Pu, Y., Zeilinger, M., Jones, C.: 'Complexity certification of the fast alternating minimization algorithm for linear MPC', *IEEE Trans. Autom. Control*, 2017, **62**, (2), pp. 888–893
[22] Wendell, R.: 'A preview of a tolerance approach to sensitivity analysis in linear programming', *Discrete Math.*, 1982, **38**, (5), pp. 121–124
[23] Hillier, F., Liberman, G.: '*Introduction to operations research*' (McGraw-Hill, New York, USA, 2001, 7th edn.)
[24] Nikandrov, A., Swartz, C.: 'Sensitivity analysis of LP-MPC cascade control systems', *J. Process Control*, 2009, **19**, (5), pp. 16–24
[25] Wendell, R.: 'The tolerance approach to sensitivity analysis in linear programming', *Managment Science*, 1985, **31**, (5), pp. 564–578