Turkish J. Eng. Env. Sci.
31 (2007) , 199 – 210.
© TÜBİTAK

Minimization Problem in Model Predictive Control Using a Modified Simplex Method

Mokeddem DIAB
Department of Electrical Engineering, University of Setif, 19000 ALGERIA
e-mail: mokeddem_d@yahoo.fr
Khellaf ABDELHAFID
Department of Electrical Engineering, University of Setif, 19000 ALGERIA
e-mail: ah_khellaf@yahoo.fr

Received 01.11.2006

Abstract

Since the simplex method requires the polyhedron to be in the positive domain, the 1-norm minimization problems are formulated by substantially increasing the size of the linear programming (LP) problems. This paper presents a simple modification that enables the simplex method to be directly applicable to a polyhedron, which extends into the negative domain. That is, instead of requiring the problem to change, the method is changed to fit the problem. The modification eliminates the need to increase the size of the problem and thus eliminates the associated computational effort.

The proposed method skips iterations and Phase 1 of the simplex method. Its computational advantage is verified in 2 examples.

Key words: Simplex method, Minimization of absolute value, 1-Norm minimization, Predictive control.

Introduction

The minimization of a sum of absolute deviations (1-norm) arises in a number of fields and applications, e.g., statistical modeling, design, control, image resolution, and signal processing. This objective is met by formulating the problem as a linear programming (LP) problem. To minimize the absolute deviations, one needs to consider both the positive and the negative values of the deviations. This means that the polyhedron (sometimes called polytope) to be searched lies in both the positive and the negative domains. However, the simplex method requires the variables to be nonnegative, that is, it requires the polyhedron to be within the positive domain.

To meet this requirement of the simplex method, the 1-norm minimization problems are formulated in the following manner:

1. By introducing new variables, minimizing their sum, and requiring that both positive and negative values of the deviations remain less than or equal to the new variables (Maciejowski, 2002; Boyd and Vandenberghe, 2004). In this formulation, the equations relating the deviations with the other optimization variables (constraints) become doubled in number.

2. By replacing each deviation by a difference of 2 nonnegative variables and then minimizing a sum of these nonnegative variables (Chang and Seborg, 1983). In this formulation, the optimization variables representing the deviations become doubled in number. One can see that to meet the requirement of the simplex method both of the conventional formulations mentioned above require a substantial increase in the size of the LP problem, that is, they require the search to be conducted in a significantly higher dimensional polyhedron. Note that the number of iterations required in solving an LP problem with m constraints and n variables through the sim-

plex method usually varies between m and $3m$, the average being $2m$ (Reklaitis et al., 1983). Moreover, the average computational effort (measured in terms of multiplications and divisions required) approximately equals $2m(nm - m^2 + n + m)$. Therefore, both of the conventional formulations require a significantly higher computational effort as compared with a formulation that does not require an increase in the size of the LP problem.

We find that it is possible to have such a formulation. In other words, instead of requiring the problem to change; the method can be changed to fit the problem; and we find that there are other opportunities for reducing the computational effort. Therefore, this paper presents a modified simplex method for the solution, which incorporates the following 3 objectives:

1. To enable the simplex method to find the optimal point in a polyhedron that extends into the negative domain, and thus avoid the increase in computational effort faced by the conventional formulations.

2. To skip the unnecessary iterations involved in the search, and thus avoid any unnecessary computational effort.

3. To formulate the problem such that an initial basic feasible solution is readily available, and thus avoid the computational effort involved in Phase 1 of the simplex method.

The method is presented for the 1-norm minimization problem as it arises in model predictive control (MPC) and can be adapted to other applications. Although the method is explained by making reference to the simplex tableau, the full tableau is not needed in performing the calculations. The method can find the same solution as obtained by using the conventional formulations. Moreover, the main modifications presented in the paper can also be used in problems other than 1-norm minimization problems.

In MPC, an optimization problem needs to be solved at every control instant (time step). The formulations of the optimization problem using LP have been proposed in the literature (Chang and Seborg, 1983; Genceli and Nikolaou, 1993; Dave et al., 1997; Rao and Rawlings, 2000). Since the on-line computational effort may be significant for certain formulations, several methods have been proposed where the solution is obtained off-line and stored (Seron et al., 2000; Bemporad et al., 2002; Kerrigan and Maciejowski, 2002; Johansen and Grancharova, 2003). Then, depending on the current condition, the stored solution is recalled and used. Since the general programming software is often unable to take advantage of the special feature of each application, Wright (1997) has suggested that optimization algorithms be customized to the applications. In this paper, the computational advantage of the proposed method over commonly used formulations is verified by its application in 2 example problems for the calculation of control moves.

Comparison of Simplex and Interior Point Methods

The simplex method starts from a basic feasible solution and moves along the boundary of the feasible region until an optimum is reached. At each step, the algorithm brings only one new variable into the basic set, regardless of the total number of variables. Thus, for problems with a large number of variables, the method may take many steps before terminating.

This behavior of the simplex method motivated researchers to develop another class of methods known as interior point methods for solving LP problems. As the name implies, in these methods, one starts from an interior feasible point and takes appropriate steps along descent directions until an optimum is found. The following figure contrasts the approach used in the simplex method with the one used in the interior point methods.

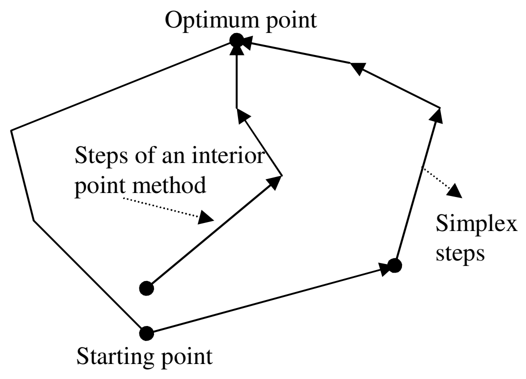

Figure 1. Typical search paths in the simplex and interior point methods.

Although the interior point method is appealing because of its superior theoretical convergence characteristics, in practice the simplex method is still more widely used.

An Introduction to the Modified Simplex Method

In this section we introduce the strategies used in meeting the first 2 objectives. The third objective of finding an initial basic feasible solution can be easily achieved and the method for doing so is described in the section “Solution of the Optimization Problem”.

To enable the simplex method to find the optimal point in a polyhedron that extends into the negative domain

The basic strategy to accomplish this objective is as follows. Use the simplex method in the usual manner on the section (portion) of the polyhedron that lies in the positive domain to locate the optimal point in this section. Then, if the search needs to go in the negative domain, flip the section of the polyhedron that is of interest into the positive domain and continue the search. The flipping can be accomplished by switching the sign of an optimization variable through redefinition. The slack variables are not allowed to switch. A record is kept of the switch status of the optimization variables to report their correct sign at the end of the solution.

Simplified Example 1 The purpose of this example is to illustrate the above strategy graphically. To do so, we assume that the polyhedron can be expressed in terms of the deviations. This assumption is not required in the proposed method where the polyhedron is expressed in terms of the deviations and the control moves. Let us consider the following objective function:

$$\text{Minimize } J = |e_1| + |e_2| \tag{1}$$

and assume that the polyhedron is a parallelogram. This parallelogram in the $e_2 - e_1$ plane may be as shown in Figure 2(a). The contours of the objective function are shown by the dotted lines and the optimal point is labeled Point 2. Let us assume that the search is to start from Point 1. For this position of the parallelogram, the optimal point can be found by searching only the section of the parallelogram that lies in the positive domain (the first quadrant). The sections that are in the negative domain (the third and fourth quadrants) can be ignored, and the search can move from Point 1 to Point 2 in one iteration.

Now consider another position of the parallelogram as shown by the solid lines in Figure 2(b). In this case the optimal point is labeled Point 3 and is not in the first quadrant. Again, let us assume that the search is to start from Point 1. The search in the first quadrant will go to Point 2 in one iteration. At this iteration, the parallelogram can be flipped around the e_1 axis. The resulting position of the parallelogram is shown by the dashed lines. Now the search can be continued in the first quadrant from Point 2 to Point 3′. Note that we do not need to search the section of the parallelogram that now lies in the second quadrant. Therefore, there is no need to flip this section into the first quadrant. At this time the sign of the e_2 coordinate of Point 3′ can be switched back to report Point 3 as the solution. The switching of Point 3′ to Point 3 is not an iteration because an iteration involves the selection of pivot column and pivot row and then the pivoting operation.

We will formulate an initial basic feasible solution such that at least a portion of the polyhedron is brought into the positive domain for the search to start. The procedure for determining when and how to flip a section of the polyhedron that lies in the negative domain is described in the sub-section “When and How to Flip the Polyhedron”.

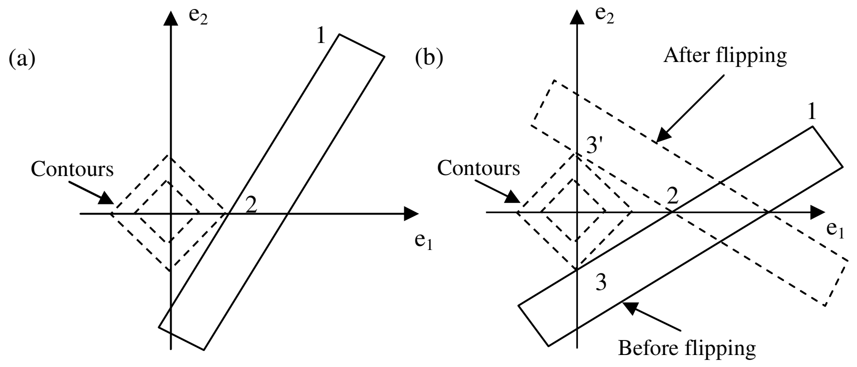

Figure 2. Movement of search (a) Point 1 to Point 2, (b) Point 1 to Point 2, Point 2 to Point 3′ and then switch Point 3′ to Point 3.

To skip the unnecessary iterations involved in the search.

This section introduces the basic strategy to accomplish the above objective. For the situation discussed in Figure 2(b) above, the search moves from Point 1 to Point 3 in 2 iterations. One may ask: Is it possible to go from Point 1 to Point 3 in one iteration? In other words, is it possible to skip the middle iteration and avoid the flipping of the polyhedron? The answer to this question is yes. To do this, we need to take a larger step than normally taken and allow e_2 to become negative temporarily. This means that we have to modify the method used for selecting the pivot rows. For a high dimensional problem, one can skip (jump over) several such iterations simultaneously before flipping the polyhedron to continue the search.

While skipping iterations, one needs to ensure that the search does not jump over an optimal point in its path. To see this consider Figure 2(a) again. While moving from Point 1 towards Point 2, the search should not jump over Point 2. Otherwise, cycling may occur. The procedure for selecting the pivot rows is described in the sub-section "Selection of the Pivot Row".

Optimization Problem

The proposed solution is applicable to optimization problems where the objective is to minimize the error in controlled variables subject to constraints on the manipulated variables. A model predictive control algorithm (MPC) is used where the error is minimized at one point P steps ahead on the prediction horizon. P is a tuning parameter and changing its value is similar to changing the move suppression factor in the dynamic matrix control (DMC) algorithm. By increasing (decreasing) the value of P, a slower (faster) response can be obtained. The value of P can be different for each of the controlled variables. The controlled performance and robustness of the MPC algorithm have been shown to be similar to that of the DMC algorithm in many cases.

The objective function considered for minimization is given below. This objective function has also been considered by Maciejowski (2002) and Chang and Seborg (1983).

$$J = \sum_{i=1}^{N_y} q_i \sum_{j=H_w}^{H_p} |e_i(k+j)| \tag{2}$$

where
$e_i(k+j)$ = predicted error in the i^{th} controlled output at j steps ahead
H_p = prediction horizon
H_w = starting point on prediction horizon for minimization of error
N_y = number of controlled outputs
q_i = weight on the i^{th} controlled output
k = current control instant.

The number of coincidence points (the points at which the error is minimized) on the prediction horizon for each of the outputs is given by

$$N_p = H_p - H_w + 1 \tag{3}$$

The total number of coincidence points, N_{pt}, and the total number of control moves into the future,N_{mt}, are given by

$$N_{pt} = N_p \times N_y \tag{4}$$

$$N_{mt} = H_u \times N_u \tag{5}$$

where H_u is the control horizon and N_u is the number of control inputs (manipulated variables).

The predicted error vector to be minimized can be written as

$$e = A\Delta u - e_c \tag{6}$$

where
A = $N_{pt} \times N_{mt}$: dynamic matrix
Δu = N_{mt}: dimensional vector of control moves
e_c = N_{pt} : dimensional vector of difference between the set points and the free response due to past inputs (as used in the DMC algorithm).

Two forms have been used in the literature for writing the dynamic matrix A (Seborg et al., 2004). Any of these forms may be used as long as the other vectors in Eq. (6) are consistent with it. Equation (6) can be rearranged as follows to bring the unknowns to the left-hand side:

$$-A\Delta u + e = -e_c \tag{7}$$

The magnitude and velocity (rate) constraints on the control inputs and constraints on the controlled and other outputs are considered. The constraints on the controlled and other outputs can be expressed in terms of constraints on the control moves by using the relationships between them. It is assumed that the current value of controls is in a feasible region since the process is already there. Then all of these constraints can be expressed through the following matrix equation:

$$C\Delta u \leq b \tag{8}$$

where the b vector is non-negative. By introducing slack variables, Eq. (8) can be written as

$$C\Delta u + s = b \tag{9}$$

where s is a vector of slack variables. Thus the optimization problem to be solved at any control instant is to minimize the objective function in Eq. (2) subject to the constraints given in Eqs. (7) and (9). The optimization variables are Δu's and e's. Different names have been used in the literature for the variables to be optimized, such as design variables and decision variables.

Solution of the Optimization Problem

In this section we describe the modified simplex method to solve the above optimization problem. A summary of the main steps in the proposed solution is presented in the sub-section "Summary of the Main Steps in the Proposed Solution". The initial basic feasible solution is formed at the current value of controls by setting Δu = 0. If the right-hand side of any equation in matrix Eq. (7) is negative, it is made positive by multiplying the equation by -1. The coefficient of the e term changed by the multiplication is set back to 1 and its switch status is changed from 1 to -1. (The switch status for the optimization variables at any time is either 1 or -1). Now we have an initial basic feasible solution where the e's of Eq. (7) and slack variables of Eq. (9) are basic variables and Δu's are non-basic variables. Note that an initial basic feasible solution can be found easily. Therefore, Phase 1 of the 2-phase simplex method is not needed. Moreover, the initial solution formed above ensures that at least a portion of the polyhedron is brought into the positive domain for the search to start. The starting value of the objective function is calculated by summing the above nonnegative values of e's after these are multiplied by their weight q from Eq. (2). To start the search, the constrained derivatives of the objective function with respect to the nonbasic variables can be calculated by using the coefficients from the starting simplex tableau and the weights q's. We present a general expression for doing so, because this expression will be used later in deciding when to flip the polyhedron. The constrained derivatives of objective function J with respect to the nonbasic variables at any iteration of the simplex method are given by the following equation (Beightler et al., 1979):

$$v_i = c_i - \sum_{j=1}^{N_b} c_j\,\alpha_{ji}, i = 1, 2, \ldots, N_{nb} \tag{10}$$

where

v_i : constrained derivative of J with respect to the i^{th} nonbasic variable

c_i: i^{th} cost coefficient in the objective function

α_{ji} : coefficient in simplex tableau for j^{th} basic variable and i^{th} nonbasic variable

N_b: number of basic variables

N_{nb}: number of nonbasic variables

The first term on the right-hand side of the above equation comes from the partial derivative and represents the direct effect of perturbations in a nonbasic variable on objective function J. These perturbations require the basic variables to change so that the constraint equations remain satisfied. The changes in basic variables affect J. The summation term adds this effect for each of the basic variables. The negative sign is present because in writing an expression for a basic variable the other α's will need to be moved to the right-hand side of the tableau. Note that the constrained derivatives are switched in sign before being placed in the objective row of the simplex tableau.

Now the iterations can be started in the usual manner to find the optimal point in the section of the polyhedron that lies in the positive domain. The above method can be used in a problem of any dimension. Before we describe when and how to flip a section of the polyhedron that lies in the negative domain, we first show it in an example. The dimension of the coordinate system in which the search moves equals the number of optimization variables considered. Therefore, to illustrate the solution graphically on a 2-dimensional sheet of paper, we consider the following simplified situation.

Simplified Example 2

Let us consider a SISO case where the control horizon $H_u = 1$ and there are lower and upper bounds on the control move Δu. The control move is to be calculated by minimizing the absolute value of the predicted error at a single point P steps ahead. In this case, the optimization problem (Eqs. (2), (7) and (8)) can be written as

$$\text{Minimize } J = |e| \tag{11}$$

subject to

$$-a_P \Delta u + e = -e_c \tag{12}$$

$$\Delta u_{\min} \leq \Delta u \leq \Delta u_{\max} \tag{13}$$

where the values of $\Delta u_{\max}$ and $\Delta u_{\min}$ are updated at every control instant using the narrower range resulting from the 2 types of constraints. a_P is the PP^{th} step response coefficient. Let us assume that the right-hand side of Eq. (12) has already been made nonnegative by using the procedure mentioned before this sub-section. Now an initial basic feasible solution (value of e) can be found by setting $\Delta u = 0$ in Eq. (12). Equation (12) imposes an equality constraint. Therefore, the polyhedron to be searched collapses into a line segment, which can be examined by plotting Eq. (12) on an e versus Δu graph. When the slope of this line is negative, the line may appear as shown in Figure 3(a). The contours of the objective function are horizontal lines, which are not shown. In this case, the minimum value of e occurs in the positive domain (the first quadrant) at $\Delta u = \Delta u_{\max}$ and there is no need to search the line segment that lies in the negative domain (the second quadrant). The optimal solution can be found by making one iteration of the simplex method where Δu will increase from zero to $\Delta u_{\max}$. The start and end points in Figure 3(a) show the movement of the search along the line segment. When the slope of the line represented by Eq. (12) is positive, it may appear as shown in Figure 3(b). In this case, the minimum value of e occurs in the negative domain (the second quadrant) at $\Delta u = \Delta u_{\min}$. The starting solution ($\Delta u = 0$) is optimal if only the positive section of the line segment is searched. However, since $\Delta u = 0$, we need to consider its negative value. This can be accomplished by flipping the line segment horizontally (around the vertical axis). The resulting situation is shown in Figure 3(c). Note that the new limit on Δu in the first quadrant is $-\Delta u_{\min}$, which is positive. Now the search can move in one iteration from the start point to the end point shown in Figure 3(c). Since the sign of the variable Δu was switched once during the search, the sign of the Δu resulting from the simplex method will need to be switched back to a negative value. As seen, only one of the limits on $\Delta u(\Delta u_{\max}$ or $\Delta u_{\min})$ is needed at any time. Therefore, only the limit $\Delta u_{\max}$ will be included in Eqs. (8) and (9). The other limit will be brought into consideration through the same equation when needed. This will cut the number of constraints on Δu (the velocity constraints) in half.

When and how to flip the polyhedron

We now return to the solution of the optimization problem in Eqs. (2), (7), and (9) and describe when and how to flip a section of the polyhedron that lies in the negative domain.

The search conducted in the section of the polyhedron that is in the positive domain can stop at a point where one or more of the optimization variables (Δu's and e's) are zero. In a non-degenerate case, this can occur when one or more of the optimization variables are nonbasic variables. We then need to determine if objective function J can be decreased by switching the sign of these variables. This can be done by calculating the new values of their constrained derivatives that will result after the switch. If one of these new constrained derivatives switches

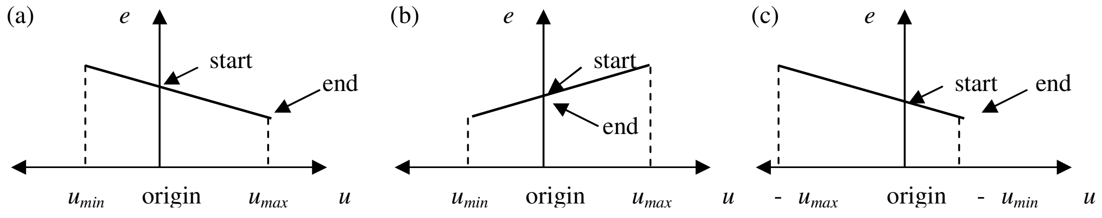

Figure 3. Movement of search when (a) optimum is in first quadrant, (b) optimum is in second quadrant, (c) optimum is brought into first quadrant.

sign, then objective function J could be decreased, otherwise the search can be stopped. In a degenerate case, variables can be exchanged before calculating their new constrained derivatives or degeneracy may be avoided in other ways.

We first consider the calculation of the new constrained derivative of objective function J with respect to an optimization variable e. If we switch an e, the second term in the right-hand side of Eq. (10) also switches in sign because of the changed alphas. However, the first term remains unchanged because of the absolute value sign in the objective function. We still want to minimize the value of e in its changed form. Therefore, the new value of vi is given by

$$v_{inew} = c_i + \sum_{j=1}^{N_b} c_j \alpha_{ji}, i = 1, 2, \ldots, N_{nb} \tag{14}$$

By adding and subtracting c_i on the right-hand side, the above expression can be written in terms of the current value of v_i as follows:

$$v_{inew} = 2c_i - v_{icurrent}, i = 1, 2, \ldots, N_{nb} \tag{15}$$

Therefore, the new value of the constrained derivative that will result after the switching can be easily predicted from the above equation. This equation can also be written in terms of the coefficients in the objective row.

$$\begin{aligned}&\alpha(nRow, i)_{\text{new}} = -2c_i - \alpha(nRow, i)_{\text{current}},\\ &i = 1, 2, \ldots, N_{nb}\end{aligned} \tag{16}$$

where $nRow$ = objective row

For the situation in Figure 2(a), the new constrained derivative with respect to e_2 stays positive and the search stops at Point 2. This can be checked by flipping the parallelogram around the e_1 axis. However, for the situation in Figure 2(b), the new constrained derivative becomes negative after flipping the parallelogram and the search continues from Point 2 to Point 3′. The cost coefficients of Δu terms in the objective function are zero. Therefore, for the new constrained derivative of objective function J with respect to a Δu, the first terms in the right-hand side of Eqs. (10), (14), (15), and (16) drop out. Moreover, Eq. (15) indicates that whenever a Δu is a nonbasic variable the new constrained derivative with respect to this variable will always switch in sign. In such cases, the decision will always be to flip the polyhedron. This situation is described in the sub-section "Simplified Example 2".

Once the decision to switch an optimization variable e or Δu is made, the polyhedron can be flipped by reversing the sign of coefficients of the corresponding column in all rows except the objective row. The coefficient for the objective row is calculated from Eq. (16). If a Δu term is being switched, the row containing velocity constraints for this variable needs to be reset. The coefficient in the column for the Δu is set back to 1 and the value in the right-hand column is switched from its current value (toggled between $\Delta u_{\max}$ and -$\Delta u_{\min}$) as indicated in the sub-section "Simplified Example 2".

Selection of the pivot row

At any iteration in the simplex method, a pivot column is selected. Then the rows (constraints) that are in the path of the current move are examined to find the steps that the rows allow. The steps are calculated by taking ratios of coefficients in the right-hand column to coefficients in the pivot column. The row that allows the smallest step is then selected as the pivot row to prevent its basic variable from becoming negative. As indicated in the section "An Introduction to the Modified Simplex Approach", we can allow the optimization variables to become negative temporarily so that we can take larger steps and skip unnecessary iterations. In this study, only the e's were allowed to become negative in this manner. As mentioned in the sub-section "To Enable the Simplex Method to Find the Optimal Point in a Polyhedron That Extends into the Negative Domain", we are using only one side of the velocity constraints at any time to cut down the number of these constraints by half. Therefore, the Δu's were not allowed to become negative by skipping their rows to avoid complexity in the required adjustments. The e's that are allowed to become negative temporarily (by skipping their row) will need to be made positive by redefinition at the end of the iteration to bring the relevant sections of the polyhedron into the positive domain. This redefinition will contribute to an increase in the value of objective function J and will offset the decrease produced in J due to the pivoting operation. To select the pivot row in the proposed method, the rows are sorted in an ascending order with respect to the steps that are calculated in the simplex method. The rows are then examined one at a time to check if a row can be skipped. This examination is continued until a row is encountered that should not be skipped. This row is then selected as the pivot row. A row may be skipped only if both of the following conditions are met.

- The basic variable for the row is a deviation, e
- $J_{next} < J_{current}$

where $J_{current}$ and J_{next} represent the final values of J that will result by pivoting the current and the next row, respectively. For calculating the final value of J for a row under examination, we need:

- The decrease in J that will result from pivoting this row. This is the usual calculation made at each iteration in the simplex method.
- The increase in J resulting from all of the rows that this row skips (jumps over).

The amount, Δe, by which a deviation, e, becomes negative if its row is skipped is given by

$$\Delta e = (\text{step\_taken} - \text{step\_skipped}) \times \text{slope} \quad (17)$$

where slope is the coefficient in the skipped row of the pivot column. Note that the slope represents the change in the basic variable for a unit change in the nonbasic variable. The validity of this expression can be checked by examining a simplex tableau. The increase in J that this skipped row causes equals $2c\Delta e$, where c is cost coefficient of the deviation in the objective function. For example, if $c = 1$, switching back an e from −5 to 5 would increase J by 10 ($2*1*5 = 10$).

As mentioned above, the right-hand column for each of the basic variables needs to be nonnegative at the end of an iteration. Therefore, after the pivoting operation, the following adjustment is needed. If an e in the j^{th} column was allowed to become negative, all coefficients in the row for this e are reversed in sign except the coefficient in the j^{th} column (which needs to be maintained = 1). The switch status for this e is also changed. Note that the above procedure is the same as mentioned in the formulation of the initial basic feasible solution. From Eq. (10), we see that this reversal in sign will affect the value of the constrained derivatives. Therefore, the coefficients in the objective row are adjusted through the following assignment statement:

$$\begin{aligned}&\alpha(nRow,i) - \alpha(nRow,i) + \sum_j c_j\,\alpha_{ji},\\ &i = (1,2,\ldots,N_{nb}) \text{ and } nCol\end{aligned} \quad (18)$$

where $nCol$ = right-hand column

The second term in the right-hand side of the above equation is multiplied by 2 because correction = 2 times the error. The summation in the above equation is carried over those rows whose e's were allowed to become negative.

Summary of the main steps in the proposed solution

a. Formulate an initial basic feasible solution.

b. Pick the pivot column containing the largest α in the objective row.

c. Pick the pivot row by using Conditions (a) and (b) in the sub-section "Selection of the Pivot Row".

d. Perform the pivot operation.

e. Switch back the e's that became negative and use Eq. (18).

f. Is one of the α's in the objective row positive?

   If yes, pick the largest α and go to Step (c).

   If no, use Eq. (16) to check if an α can be made positive by switching a nonbasic variable (e or Δu).

   If yes, pick the largest α, flip the polyhedron, and go to Step (c).

   If no, stop.

Performance

At any control instant, the task of the optimization algorithm is to compute the control moves based upon the e_c vector supplied to it. A time-based run, e.g., a response to a step change in set points, may not force the optimization algorithm to test values of e_c in all of the regions around the origin. Moreover, a time-based run involves other computations, e.g., the current outputs, the predicted trajectories due to past inputs, and bias terms for feedback. These additional computations will interfere in studying the computational effort of the optimization algorithm. Therefore, to test the performance of the proposed optimization algorithm, we supplied different values of the e_c vector to it directly through Eq. (7) and computed the control moves. The values of e_c

were supplied so that they covered all of the regions around the origin. Both magnitude and velocity constraints on the control inputs are considered in the example problems.

The control moves were also computed by using the following 2 commonly used formulations that convert the optimization problem in Eqs. (2), (7), and (8) into an LP problem. The LP problem was then solved by using the simplex method:

1. LP formulation by increasing the number of constraints: In this formulation, the e vector is replaced by another vector, γ. The absolute values are then handled by replacing Eq. (6) by the following 2 equations where the number of these constraints becomes doubled:

$$\gamma \geq (A\Delta u - e_c) \tag{19}$$

$$\gamma \geq -(A\Delta u - e_c) \tag{20}$$

The Δu vector is replaced by a nonnegative Δû vector by using the following expression:

$$\Delta u \equiv \Delta \hat{u} + \Delta u_{\min} \tag{21}$$

Then only the right-hand side of the constraints on these variables is needed as follows:

$$\Delta \hat{u} \ \leq \Delta u_{\max} - \Delta u_{\min} \tag{22}$$

Since the number of constraints in this LP formulation is more than the number of optimization variables, the dual of the LP problem was solved. As expected, the dual formulation required less computational effort. This dual LP formulation is referred to as Algorithm 2 (Yash, 2004) in this paper and its results are reported.

2. LP formulation by increasing the number of variables: In this formulation, the optimization variables e's are replaced by a difference of 2 nonnegative variables in Eq. (7) and are thus doubled in number. A sum of these nonnegative variables is then minimized. The Δu vector is replaced by a nonnegative Δû vector in all equations as in the previous formulation. This formulation is referred to as Algorithm 3 (Yash, 2004) in this paper.

All 3 optimization algorithms were programmed using Matlab. The control moves calculated by the proposed algorithm for implementation were the same as those obtained by the other algorithms. However, the computational effort required was different. Since the information on the number of floating-point operations required was unavailable in Matlab, a rough estimate of the computational effort of the algorithms is given by reporting the number of iterations and the computational times taken in reaching the optimal solution.

The performance of the proposed algorithm is presented in 2 example problems for the following 3 tuning configurations:

1. $H_u > 1$, $N_p > 1$, $H_w = 1$
2. $H_u = 1$, $N_p > 1$, $H_w = 1$
3. $H_u = 1$, $N_p = 1$, $H_w > 1$

When the control horizon $H_u = 1$, the magnitude and velocity constraints on the control inputs can be combined into the same equation and expressed as follows:

$$\Delta u_{\min} \leq \Delta u \leq \Delta u_{\max} \tag{23}$$

This can be done by calculating the narrower range of Δu's at every control interval that results from the 2 types of constraints, and then supplying the updated values of $\Delta u_{\max}$ and $\Delta u_{\min}$ to the optimization algorithm. This helps in reducing the number of constraints in the optimization problem. Therefore, the second tuning configuration above was included in the study. In addition, since the single prediction controller provides further reduction in the size of the optimization problem, the third tuning configuration above was also studied.

Example Problem 1

In this example, a pilot-scale distillation column (Wood and Berry, 1973) is considered, where the open-loop transfer function between the control inputs and the controlled outputs is

$$G(s) = \begin{bmatrix} \frac{12.8\,e^{-s}}{16.7s+1} & \frac{-18.9\,e^{-3s}}{21s+1} \\ \frac{6.6\,e^{-7s}}{10.9s+1} & \frac{-19.4\,e^{-3s}}{14.4s+1} \end{bmatrix} \tag{24}$$

Five values of set points (origin + 2 values on each side of the origin) for each of the output variable were considered. This resulted in 25 ($5^2 = 25$) sets of the e_c vector. In these sets, the values of e_c for a particular output over the prediction horizon were the same as its set point because the process was assumed to be at steady-state. Control moves were calculated for each of the 25 sets and each set represented a run. The average number of iterations for a run was calculated by dividing the total number of iterations by 25. Similarly, the average computational time for a run was calculated by dividing the total time by 25. The values of set points, constraints, and other parameters are as follows:

$$T = 1min N = 100 |\Delta u| \leq 0.05 |u| \leq 0.15 q's = 1$$

Five values of set points = (1 0.5 0 -0.5 -1)

The average numbers of iterations for a run are shown in Table 1 and the average computational times for a run are shown in Table 2. The proposed algorithm provides a large reduction in the number of iterations and in the computational time in the first 2 cases. In the case of the single prediction controller, since the number of iterations was very small, there was no appreciable difference in the results of the 3 algorithms.

Example Problem 2

In this example, the "Shell" heavy oil fractionators (Prett and Morari, 1987) are considered, where the open-loop transfer function between the control inputs and the controlled outputs is

$$G(s) = \begin{bmatrix} \frac{4.05\,e^{-27s}}{50s+1} & \frac{1.77\,s^{-28s}}{60s+1} & \frac{5.88\,e^{-27s}}{50s+1} \\ \frac{5.39\,e^{-18s}}{50s+1} & \frac{5.72\,e^{-14s}}{60s+1} & \frac{6.90\,e^{-15s}}{40s+1} \\ \frac{4.38\,e^{-20s}}{33s+1} & \frac{4.42\,e^{-22s}}{44s+1} & \frac{7.20}{19s+1} \end{bmatrix} \qquad (25)$$

Three different values of set points (origin + one value on each side of the origin) for each of the 3 output variables were considered. This resulted in 27 ($3^3 = 27$) different sets of the e_c vector. The calculations were performed as described for Example Problem 1. The values of set points, constraints, and other parameters are as follows:

$$T = 4min N = 65 |\Delta u| \leq 0.2 |u| \leq 0.5 q's = 1$$

Three values of set points = (0.5 0 -0.5)

The average numbers of iterations for a run are shown in Table 3 and the average computational times for a run are shown in Table 4. Again, the proposed algorithm provides a large reduction in the number of iterations and in the computational time in the first 2 cases. In the case of the single prediction controller, since the number of iterations was very small, there was no appreciable difference in the results of the 3 algorithms.

Table 1. Average number of iterations for a run (Example Problem 1).

Tuning parameters	Proposed Algorithm	Algorithm 2	Algorithm 3
$H_u = 20$ $N_p = 100$ $H_w = 1$	135	450	520
$H_u = 1$ $N_p = 50$ $H_w = 1$	9	118	152
$H_u = 1$ $N_p = 1$ $H_w = 5$	2	3	3

Table 2. Average time for a run in milliseconds (Example Problem 1).

Tuning parameters	Proposed Algorithm	Algorithm 2	Algorithm 3
$H_u = 20$ $N_p = 100$ $H_w = 1$	932	9598	12430
$H_u = 1$ $N_p = 50$ $H_w = 1$	16	474	553
$H_u = 1$ $N_p = 1$ $H_w = 5$	1.1	1.1	1.2

Table 3. Average number of iterations for a run (Example Problem 2).

Tuning parameters	Proposed Algorithm	Algorithm 2	Algorithm 3
$H_u = 10$ $N_p = 65$ $H_w = 1$	185	531	614
$H_u = 1$ $N_p = 50$ $H_w = 1$	35	184	243
$H_u = 1$ $N_p = 1$ $H_w = 10$	3	5	4

Table 4. Average time for a run in milliseconds (Example Problem 2).

Tuning parameters	Proposed Algorithm	Algorithm 2	Algorithm 3
$H_u = 10$ $N_p = 65$ $H_w = 1$	912	9915	10946
$H_u = 1$ $N_p = 50$ $H_w = 1$	83	1563	1830
$H_u = 1$ $N_p = 1$ $H_w = 10$	1.6	1.6	1.4

Comments

- Because of the round-off error, epsilons are needed in computer programs for deciding when a number should be considered as a zero. Some sensitivity in the results was observed to the values of these epsilons.

- The proposed method uses a much smaller size LP problem and skips Phase 1. One could justify its computational advantage based on these arguments. A verification of this advantage has been provided by its application in 2 example problems.

- Most modern control packages use QP based algorithms to solve the optimization problem in MPC. Since the LP and QP based MPC algorithms can be tuned to provide similar control performance (Gillis, 1998), the computational advantage offered by the proposed method could improve the suitability of the LP based algorithms for MPC. Small process control systems, common in many plants, can benefit from faster algorithms.

- As mentioned in the introduction section, the 1-norm minimization problem arises in a number of fields and applications. This methodology can also be adapted for other applications.

Conclusions

By using a simple modification, the simplex method can find the optimal point in a polyhedron that extends into the negative domain. Thus the need to increase the size of the LP problem to bring the entire polyhedron into the positive domain is eliminated. The unnecessary iterations in the search can be skipped. The modifications presented can also be used in problems other than the 1-norm minimization problem.

The modified simplex method offers a significant computational advantage over the conventional formulations for the solution of 1-norm minimization problems.

Nomenclature

a_P	change in output, P steps ahead due to a unit step change in control input
c_i	i^{th} coefficient in objective function
e	deviation or error

H_p	prediction horizon
H_u	control horizon
H_w	starting point on prediction horizon for minimization of error
J	objective function
k	current sampling instant
N	model horizon, number of control intervals in which the open-loop response settles.
N_{mt}	total number of control moves (current and future)
N_p	number of point at which the error is minimized for each output
N_{pt}	total number of coincidence points
N_u	number of control inputs (manipulated variables)
N_y	number of controlled outputs (controlled variables)
$nCol$	right-hand column
$nRow$	objective row
q_i	weight on the i^{th} controlled variable
s	slack variable
T	control interval
Δu	control move
$\Delta u_{\max}$	upper bound on the control move
$\Delta \mathrm{u}_{\min}$	lower bound on the control move
v_i	i^{th} constrained derivative of objective function

Greek letters

α	coefficient in simplex tableau
γ	new variable in LP formulation (algorithm 2)
Bold face	Indicates a vector or matrix

References

Beightler, C.S., Phillips, D.T. and D.J. Wilde, "Foundations of Optimization," Prentice-Hall, Englewood Cliffs, N.J. 74, 1979.

Bemporad, A., Borrelli, F. and M. Morari, "Model Predictive Control Based on Linear Programming-the Explicit Solution," IEEE Trans. on AC. 47, 1974-1985, 2002.

Boyd, S. and Vandenberghe, L. "Convex Optimization," Cambridge University Press, Cambridge, UK, 294, 2004.

Chang, T.S. and Seborg, D.E. "A Linear Programming Approach for Multivariable Feedback Control with Inequality Constraints," Int. J. of Control. 37, 583-597, 1983.

Dave, P., Willig, D.A., Kudva, G.K., Pekny, J.F. and Doyle, F.J. "LP Methods in MPC of Large Scale Systems: Application to Paper-Machine CD Control," AIChE J. 43, 1016-1031, 1997.

Genceli, H. and Nikolaou, M., "Robust Stability Analysis of Constrained ℓ1-norm Model Predictive Control. AIChE J., 39, 1954-1965, 1993.

Gillis, J.D., "Constrained Simplified Model Predictive Control of a Fluidised Bed Reactor," M.A.Sc. Thesis, Dalhousie University, Halifax, N.S., 1998.

Johansen, T.A. and Grancharova, A., Approximate Explicit Constrained Linear Model Predictive Control Via Orthogonal Search Tree, IEEE Trans. on AC. 48, 810-815, 2003.

Kerrigan, E.C. and Maciejowski, J.M., "Designing Model Predictive Controllers with Prioritized Constraints and Objectives," Proc. IEEE International Symposium on Computer Aided Control System Design. 33-38, 2002.

Maciejowski, J.M., "Predictive Control with Constraints," Prentice Hall, England 154, 2002.

Prett, D.M. and Morari, M., Editors. "The Shell Process Control Workshop. Process Control Research: Industrial and Academic Perspectives," Butterworths, Boston, MA, 1987.

Rao, C.V. and Rawlings, J.B., "Linear Programming and Model Predictive Control," J. Process Control. 10, 283-289, 2000.

Reklaitis, G.V., Ravindran, A. and Ragsdell, K.M., "Engineering Optimization," John Wiley, New York, N.Y., 168, 1983.

Seborg, D.E., Edgar, T.F. and Mellichamp, D.A., "Process Dynamics and Control," 2nd ed., John Wiley, New York, N.Y., 547, 2004.

Seron, M.M., De Dona, J.A. and Goodwin, G.C., "Global Analytical Model Predictive Control with Input Constraints," Proc. of 39th IEEE Conf. on Decision and Control. 1, 154-159, 2000.

Wood, R.K. and Berry, M.W., "Terminal Composition Control of a Binary Distillation Column," Chem. Eng, Sci. 28, 1707, 1973.

Wright, S.J., "Applying New Optimization Algorithms to Model Predictive Control," In 5th Intl. Conf. on Chemical Process Control, AIChE Symposium Series. 93, 147-155 1997.

Yash P.G., "Solution of Low-Dimensional Constrained Model Predictive Control Problems", ISA Transactions. 43, 499-508, 2004.