Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Extending LP Optimizer Functionality for Model Predictive Control

Willy Wojsznis, Terry Blevins, Peter Wojsznis,

Ashish Mehta

Emerson Process Management,

12301 Research Blvd., Austin, TX 78759

Willy.Wojsznis@EmersonProcess.com

KEYWORDS

Optimization, Linear Programming - LP, Quadratic Programming - QP, Model Predictive Control - MPC

ABSTRACT

The subject of this paper is linear programming (LP) optimizer application with Model Predictive Control (MPC). This extremely successful merger of two major control technologies is enabled as a consequence of the MPC feature of providing a prediction of the process outputs up to steady state, thus creating the required conditions for optimizer operation. However, the standard LP algorithm which finds a solution only within acceptable limits, does not perform properly when some of the predicted process outputs are out of limits. On the other hand, the optimizer applied with the MPC controller must always find a solution and thus there is a need to extend the original optimization formulation. This paper presents robust and reliable ways of handling optimized process outputs that are out of the limits. The technique is based on the priority structure, penalizing slack variables, and redefining the constraint model. In addition LP functionality is extended by defining one- or two-sided ranges around the control variables set points and preferred settling values for the manipulating variables. This technique has been implemented in an industrial control system and will be presented interactively by simulating the optimization and control of a distillation column.

INTRODUCTION

Optimizers are an inherent part of MPC functionality, wherein they are applied towards the economic optimization and constraint handling objectives [1]. In normal operating conditions, an optimizer provides economically optimal solution within acceptable ranges and limits. If a solution does not exist within the predefined ranges and limits, the optimizer should have the means to recover from the infeasibility. The existing recovery techniques are based on the priorities of the constrained and controlled variables. There are several known approaches to this problem. In [2], integer variables are used to cope with prioritization. The minimization of the size of the violation is performed by solving a sequence of mixed integer optimization problems. In [3], an algorithm solves a sequence of LP or QP (quadratic programming) problems in the case of infeasibility. This algorithm, like the previous one, minimizes the violations of those constraints that cannot be fulfilled. However, both approaches are computationally intensive and therefore inadequate, particularly for fast real time applications. In [4], an algorithm has been developed for off-line LP weights design in such a way that the computed constraint violations are optimal. It takes the burden of excess computations off-line but presents an additional off-line optimization problem.

Since the LP solution lies at the constraints boundary, therefore optimal steady state targets tend to bounce around [1]. LP functionality is particularly unsatisfactory when no objectives are defined for most or all of the variables. In such cases the control requirement would be to preserve process input and output values, while a standard LP optimizer will tend to drive the process variables to the boundaries. This is because the initial LP solution is at the boundaries and since there are no objectives for improving this solution, it becomes the final solution. In order to prevent such behavior QP optimization is applied, substantially increasing the complexity of the implementation.

In this paper we present a technique for constraint handling and extending LP functionality using penalized slack variables that is suitable for on-line implementation. This technique provides the functionality delivered by the QP optimizer with an insignificant increase in LP complexity. For the reader's convenience, basics of MPC optimization as outlined in [5, 6] precede the extended LP functionality presentation.

BASICS OF MPC OPTIMIZATION

For processes with several manipulated and controlled variables, optimization techniques, generally in the form of an LP, are an essential component of Model Predictive Control application. MPC uses incremental manipulated variables (MV) at the present time or sum of increments of MV over the control horizon and incremental values of controlled and constrained variables (CV) at the end of the prediction horizon, instead of positional current values as in

0-7803-9568-9/05/$20.00 ©2005 IEEE

typical LP applications. The LP technique uses a steady state model, therefore steady state condition is required for its application. With the prediction horizon normally used in MPC design, future steady state is guaranteed for self-regulating processes. Predicted process steady state equation for an *m* by *n* input-output process, with prediction horizon *p,* control horizon *c*, in the incremental form is:

$$\Delta CV(t+p)=A*\Delta MV(t+c) \quad (1)$$

where

$\Delta CV(t+p)=[\Delta cv_1,\ldots, \Delta cv_n]^T$ - the vector of the predicted changes in outputs at the end of the prediction horizon,

$$A=\begin{bmatrix} a_{11} & \ldots & a_{1m} \\ \ldots & & \ldots \\ a_{n1} & \ldots & a_{nm} \end{bmatrix}$$

is the process steady state *n x m* gains matrix,

$\Delta MV(t+c)=[\Delta mv_1,\ldots, \Delta mv_m]^T$ - the vector of the sum of the changes over the control horizon made by every controller output mv_i.

$$\Delta mv_i=\sum_{j=1}^{c}\Delta mv_i(t+j) \qquad i=1,2,\ldots,m$$

The changes in process variable values should satisfy the limits on both MVs and CVs, i.e.:

$$MV_{min}\le MV_{current}+\Delta MV(t+c)\le MV_{max} \quad (2)$$

$$CV_{min}\le CV_{predicted}+\Delta CV(t+p)=CV_{predicted}+A*\Delta MV(t+c)\le CV_{max} \quad (3)$$

The objective function for maximizing product value and minimizing raw material cost can be defined jointly in the following way:

$$Q_{min}=-UCV^T*\Delta CV(t+p)+UMV^T*\Delta MV(t+c) \quad (4)$$

where *UCV* denotes the cost vector for a unit change in CV process value and *UMV* denotes the cost vector for a unit change in MV process value. Applying (1), the objective function expressed in terms of the MV is:

$$Q_{min}=-UCV^T*A*\Delta MV(t+c)+UMV^T*\Delta MV(t+c)$$

The LP solution is always located at one of the vertices of the region of feasible solutions [7]. To find this solution, the LP algorithm calculates the objective function for an initial vertex and improves the solution every next step until it determines the vertex with minimum value of the objective function as the optimal solution. The optimal MV values are applied to the MPC controller as the target MV values to be achieved within the control horizon. If the MPC controller is squared, i.e., the number of MVs is equal to the number of CVs in the controller, the MV targets can be effectively achieved by a change in the CV set point value [8].

$$\Delta CV^C=A^C*\Delta MVT$$

ΔMVT- change in optimal target value of MV

CV^C – subset of control and constraint variables that are included in the MPC squared controller.

The optimal MV values are achieved by managing the CV^C set points. ΔCV^C represents the required change in the CV^C set point value.

In normal situations, the target set points are within acceptable ranges and constraint variables are within limits. However when disturbances are too severe to be compensated, the optimizer may not find a solution within limits and some mechanism is needed for handling infeasibilities. An effective technique for infeasibility handling and extending LP functionality is the subject of the next chapters.

INFEASIBILITY HANDLING BY USING SLACK VARIABLES

The concept is based on a special use of parameters called slack variables. In linear programming slack vectors $S_{\max}\ge 0$ and $S_{\min}\ge 0$ are used to transform the inequalities in (3) into the following equalities:

$$CV_{predicted}+A*\Delta MV(t+c)=CV_{min}+S_{min} \quad (5)$$

$$CV_{predicted}+A*\Delta MV(t+c)=CV_{max}-S_{max} \quad (6)$$

The equality is required for the linear programming model thus slack variables serve only as formal parameters with no specific application meaning. However, adding slack variables does serve to increment the degrees of freedom. Therefore, adding yet another slack variable for each equation will increase the degrees of freedom and will allow finding a solution in situations where the solution may not exist. Slack variables may be applied to extend the range limits, with the slack vector $S^+\ge 0$ for the high limit violation and the slack vector $S^-\ge 0$ applied for the low limit violation, as below:

$$CV_{predicted}+A*\Delta MV(t+c)=CV_{min}+S_{min}-S^- \quad (7)$$

$$CV_{predicted}+A*\Delta MV(t+c)=CV_{max}-S_{max}+S^+ \quad (8)$$

Figure 1 illustrates this concept of slack variables for constraint variables (process outputs with no set points). *S(i)* denotes the component *i* of the relevant slack vector. Note that there will be a pair of equations (7,8) for each value of the CV prediction. Depending on the actual prediction value,

some of the slack variables will be zero. Only the non-zero slack variables are shown in figures 1 to 4.

To get the LP solution within ranges or to minimally violate the ranges, the new slack variables should be penalized and these penalties should be significantly higher than economic costs or profits. Therefore, the objective function (4) should be extended by adding the terms $PS_-^T * S^-$ and $PS_+^T * S^+$.

$$Q_{min} = -UCV^T * \Delta CV(t+p) + UMV^T * \Delta MV(t+c) + PS_-^T * S^- + PS_+^T * S^+ \quad (9)$$

where

PS_- is the penalty vector for violating low limits

PS_+ is the penalty vector for violating high limits

$PS_- >> UCV$ and $PS_+ >> UCV$, also

$PS_- >> UMV$ and $PS_+ >> UMV$

All components of the vectors PS_- and PS_+ should be significantly larger than economic cost/profit vectors. Generally, it is reasonable to assume that the smallest component of the vectors PS_- and PS_+ is greater than the largest component of the UCV vector.

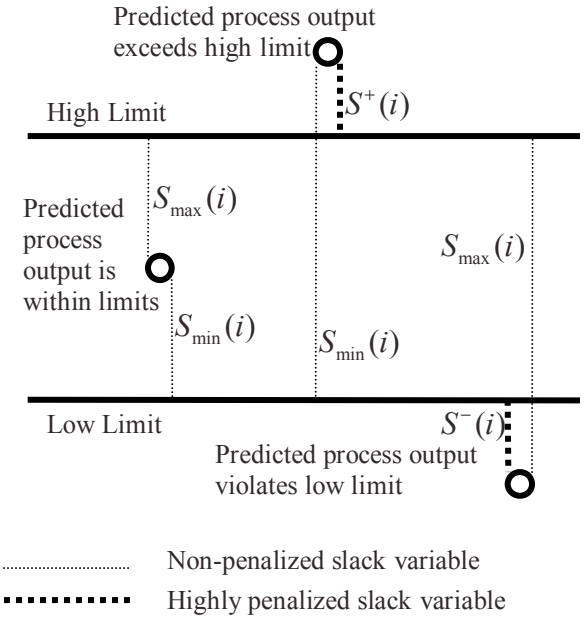


Figure 1. Application of Slack Variables for Constraint Handling

TECHNIQUE FOR EXTENDING LP FUNCTIONALITY

An extension of the slack variable application for MPC optimization is achieved by using slack variables applied to set point optimization within acceptable ranges. The ranges are defined around set points within high and low set point limits. Ranges can be single sided or two sided. Single sided ranges are associated with minimization and maximization (Figure 2) objective functions. Two sided ranges have no economic objectives other than realizing the optimal solution as close as possible to the set point within the defined ranges, as extended by the penalized slack variables (Figure 3). If the range is equal to zero, then we have a set point with the highly penalized slack variables around it.

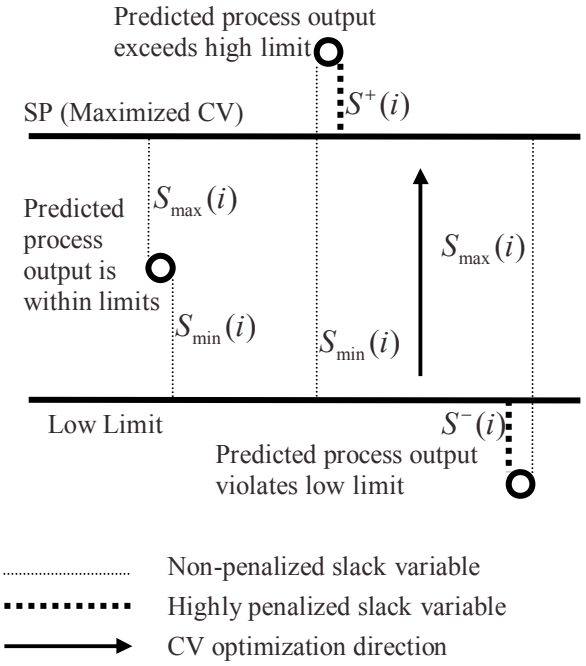


Figure 2. Slack Variables Application to Maximized Single Sided Range Control

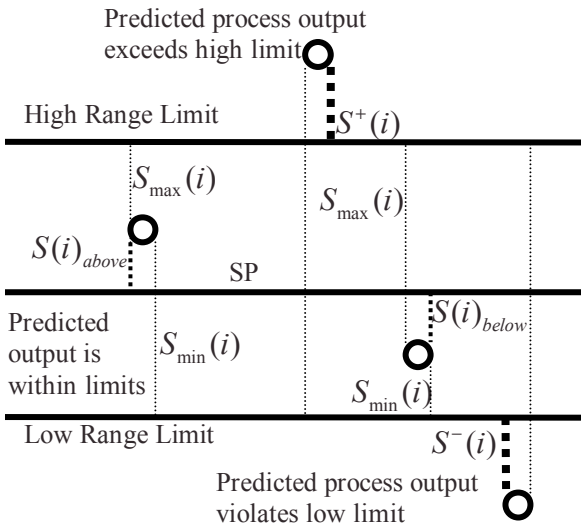


Figure 3. Slack Variables Application to Two-Sided Range Control

The equations for set point control with the two-sided ranges, for each CV prediction, are in the form:

$$CV_{predicted}+A*\Delta MV(t+c)=SP-S_{below}+S_{above}$$

$$CV_{predicted}+A*\Delta MV(t+c)=CV_{min}+S_{min}-S^{-}$$

$$CV_{predicted}+A*\Delta MV(t+c)=CV_{max}-S_{max}+S^{+}$$

S_{below} and S_{above} are additional vectors of slack variables for the solutions below and above the set points. The term

$PSP^{T}_{below} * S_{below}+PSP^{T}_{above} * S_{above}$ should be added to the objective function, where

PSP^{T}_{below} is the unit penalty for the solution below the set point , and

PSP^{T}_{above} is the unit penalty for the solution above the set point.

Applying penalized slack variables in this manner, the optimizer can always find a solution at its first execution, even if the solution is outside the output limits. This approach also allows handling process inputs in a more flexible way. Inputs (MVs) have only hard constraints. It is also possible to define soft constraints, contained within hard constraints, for some of the inputs. Introducing penalized slack variables for the soft constraints, it is easy to define a penalized range for the MVs. For this the following equations should be included into the model along with equation (2):

$$MV_{min}^{soft}-S^{-}_{softmin}=MV_{current}+\Delta MV(t+c) \quad (10)$$

$$MV_{max}^{soft}+S^{+}_{softmax}=MV_{current}+\Delta MV(t+c) \quad (11)$$

The same approach is used to define preferred settling value (PSV), the preferred value for the MV, as shown in Figure 4. PSV is defined by the user or it can be the last value of the MV.

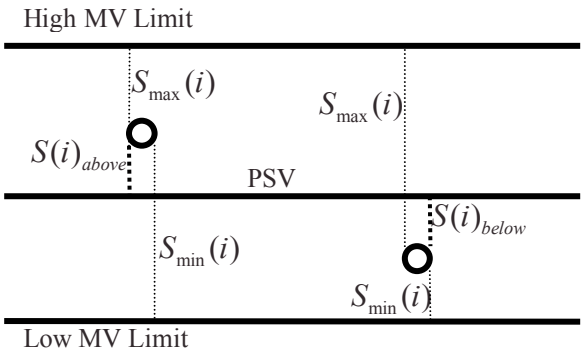


Figure 4. Using penalized slack variables to account for the Preferred Settling Value

The equations for an MV with PSV are:

$$PSV-S_{below}+S_{above}=MV_{current}+\Delta MV(t+c) \quad (12)$$

$$MV_{min}+S_{min}=MV_{current}+\Delta MV(t+c) \quad (13)$$

$$MV_{max}-S_{max}=MV_{current}+\Delta MV(t+c) \quad (14)$$

PSV is the lowest priority objective to be achieved after the other objectives – economic, constraint and/or control are satisfied. Objective function terms for the penalized MV slack vectors S_{below}, S_{above} are set in a similar way as those for the CV set point slack vectors.

IMPLEMENTATION EXAMPLE

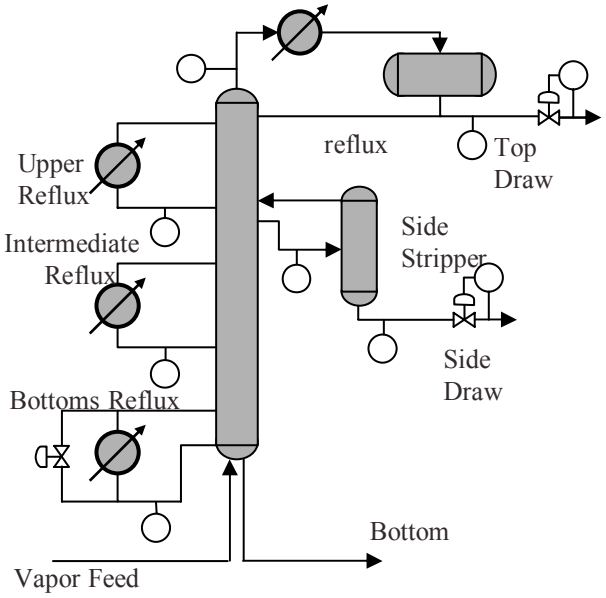
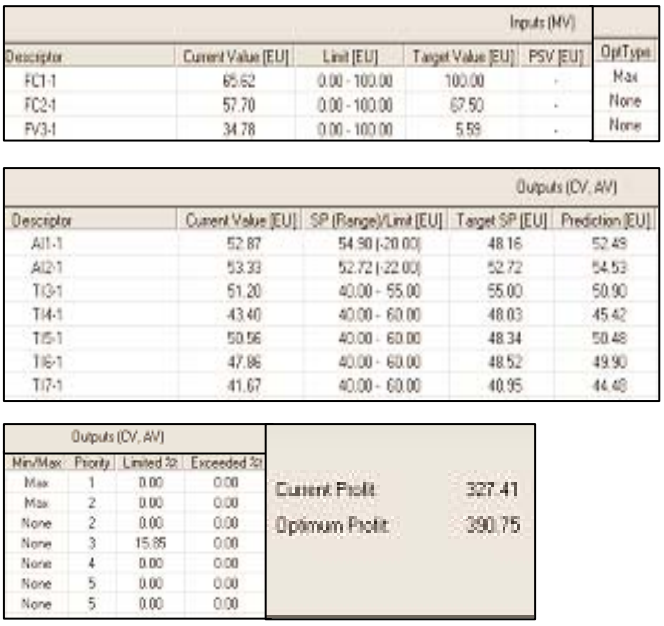


Figure 5: Shell Heavy Oil Fractionator

At a Shell Process Control Workshop in 1987, Prett and Morari proposed a generic control problem often referred to as the Shell Heavy Oil Fractionator (HOF). This problem examines a multivariable control strategy's ability to track set points, handle constraints, reject disturbances and meet economic objectives in the presence of long dead times, model gain uncertainty, multiple constraints and mixed fast and slow responses, and is often used as a reference MPC test [9, 10]. The full 5x7 process is difficult to control as model matrix lines are close to being collinear. The Shell HOF (Figure 5) has top and side draws with product specifications determined by economics and operating constraints. A bottom draw has no product specification. Heat enters the HOF with the gaseous feed and heat is removed from the column to achieve the desired specification in three side-circulating loops. The top two side-circulating loops are heat integrated with other parts of the plant. Since operational conditions in other units vary, they can be a source of disturbances on any HOF. In all test cases, step disturbances in the duties of the two loops must be rejected by any proposed control strategy. Heat removal from the bottom side-circulating loop is regulated by adjusting steam production. Since the duty is the heat returned to the fractionator, the economic incentive is to minimize the bottom reflux duty. There is an operational constraint on the temperature in the lower part of the column.

An industrial DCS system with embedded MPC controller, optimizer and modeling software is applied to the Shell problem. The optimizer, with the modifications described in this paper, is integrated with the MPC controller and implemented as a control function block executing in the controller [11]. A workstation application connects to the function block for process testing, process model development, simulation and operation. The screen captures in Figure 6 show the parameters used in the optimization. The optimizer functionality is evident from the column descriptions in the figure.

Constraints violation was achieved by applying high-level disturbances or by setting low/high limits on the inputs and outputs that are violated by their actual values. The detailed tests confirmed the assumed property of the technique; in particular the optimizer acted effectively in handling constraints and providing control functionality. The reported conclusions of the tests were that the embedded MPC formulation, containing an LP optimizer based on steady state models, met the Shell Control Problem criteria in *all test cases* [10]. The optimizer performance and functionality will be demonstrated at the interactive presentation.



Inputs (MV)

Descriptor	Current Value [EU]	Limit [EU]	Target Value [EU]	PSV [EU]	OptType
FC1-1	65.62	0.00 - 100.00	100.00	-	Max
FC2-1	57.70	0.00 - 100.00	67.50	-	None
FV3-1	34.78	0.00 - 100.00	5.59	-	None

Outputs (CV, AV)

Descriptor	Current Value [EU]	SP (Range)/Limit [EU]	Target SP [EU]	Prediction [EU]
AI1-1	52.87	54.90 (-20.00)	48.16	52.49
AI2-1	53.33	52.72 (-22.00)	52.72	54.53
TI3-1	51.20	40.00 - 55.00	55.00	50.90
TI4-1	43.40	40.00 - 60.00	48.03	45.42
TI5-1	50.56	40.00 - 60.00	48.34	50.48
TI6-1	47.86	40.00 - 60.00	48.52	49.90
TI7-1	41.67	40.00 - 60.00	40.95	44.48

Outputs (CV, AV)

Min/Max	Priority	Limited Sp	Exceeded Sp
Max	1	0.00	0.00
Max	2	0.00	0.00
None	2	0.00	0.00
None	3	15.85	0.00
None	4	0.00	0.00
None	5	0.00	0.00
None	5	0.00	0.00

Current Profit 327.41
Optimum Profit 390.75

Figure 6. Embedded MPC optimizer parameter view. Top: Input (Manipulated); Center: Outputs (Controlled and Constraint); and Bottom: Output optimization/priority and economic profit value.

CONCLUSIONS

Handling infeasible LP solutions by applying penalized slack variables is an extremely flexible and effective approach. The principles of constraint handling presented here provide an easy way to develop a number of modifications to the constraint models so as to satisfy the specific requirements for control, constraint, and manipulated variables in an MPC formulation. In essence, the approach extends LP functionality to match QP features without the additional complexity. The technique is successfully embedded into an industrial controller, providing unique MPC functionality.

REFERENCES

1. Qin, S. J. and Badgwell, T. A., "An Overview of Industrial Model Predictive Control Technology," *Fifth International Conference on Chemical Process control,* pages 232-256, AIChE and CACHE, 1997.

2. Tyler, M.L. and Morari M., "Propositional Logic in Control and Monitoring Problems," In *Proceedings of European Control Conference '97,* pages 623-628, Bruxelles, Belgium, June 1997.

3. Vada, J., Slupphaug, O. and Foss, B.A., "Infeasibility Handling in Linear MPC subject to Prioritized Constraints," In *PreprintsIFAC'99 14th World Congress,* Beijing, China, July 1999.

4. Vada, J., Slupphaug, O. and Johansen, T.A., "Efficient Infeasibility Handling in Linear MPC subject to Prioritized Constraints," In *ACC2002 Proceedings,* Anchorage, Alaska, May 2002.

5. Wojsznis, W., T., Thiele D., Wojsznis, P., and Mehta, A., "Integration of Real Time Process Optimizer with a Model Predictive Function Block," *ISA Conference*, Chicago, October 2002.

6. Wojsznis, W., Terry Blevins, Mark Nixon, Peter Wojsznis, 'Infeasibility Handing in MPC with Prioritized Constraints," *ISA Conference*, Houston, October 2003.

7. William H. Press, Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery, "Numerical Recepies in C," *Cambridge University Press*, 1997.

8. Mehta, A., Wojsznis, W., Thiele, D., and Blevins, T., "Constraints Handling in Multivariable System by Managing MPC Squared Controller," *ISA Conference*, Houston, October 2003.

9. Maciejowski J.M., "Predictive Control with Constraints," *Prentice Hall,* 2002.

10. Boudreau, M., "Squared Model Predictive Controller Performance on the Shell Standard Control Problem," *ISA Conference*, Houston, October 2003.

11. http://easydeltav.com/keytechnologies/index.asp