

# Stochastic linear model predictive control using nested decomposition

Andrew J. Felt

Department of Mathematics and Computing,  
University of Wisconsin-Stevens Point, Stevens Point, WI 54481-3897.  
afelt@uwsp.edu.

## Abstract

We begin with a traditional model predictive control problem using the  $l_1$  norm in the objective function, and then allow the model parameters to be stochastic, with discrete distributions and finite support. We apply the nested decomposition algorithm for multistage stochastic linear programming to the resulting problem. The result is an algorithm for model predictive control that features the realism of model uncertainty, the potential speed of linear objective functions, and can be implemented in parallel.

## 1 Introduction

Model predictive control is a popular strategy for control of many industrial processes. This strategy uses an open-loop, multistage model of the process to predict the effects on the output, or controlled, variables from previous and planned control moves. The goal is to find a series of future control moves that, when implemented, would bring the controlled variables as close as possible to the desired trajectory. Usually, only the control moves from the first time stage are implemented. Then a new set of process measurements are made and the model is solved again.

At the heart of model predictive control is an optimization problem that must be solved relatively quickly. The problem has multiple time stages, and is traditionally linearly constrained with a quadratic objective function. With such problems, the time stages are usually aggregated so that the problem is simply a single huge linearly constrained quadratic optimization problem.

In such a setting, when a more efficient algorithm is found, the result is that model predictive control can be used on larger problems or on problems that require faster solution. It is for this reason that some practitioners [3] have turned to linear objective functions, so that linear programming (LP) methods may be used. Solutions of linear programs were found to be much

faster than those for quadratic programs of similar size and structure.

However, Rao and Rawlings [8] found that model predictive control using a linear objective function can yield control that is either dead-beat or idle. This means that the control moves were either overly aggressive or extremely passive. The solution to the LP was all or nothing.

It is common in many applications, not just model predictive control, to see such undesirable solutions when applying linear programming. The nature of the LP is such that an optimal solution, when one exists, is always at the extreme boundary of feasibility. This “all or nothing” behavior for LP models has been observed, for example, in other fields such as finance and planning, where it has been improved upon by specifying random distributions for some of the model parameters [2, 5, 6, 7]. The resulting multistage stochastic linear programs (MSLPs) have produced solutions that are of higher quality than their deterministic counterparts. The solutions to MSLPs are, in general, less aggressive than those of deterministic LPs. The stochastic programming practitioners have found that when we admit *within our model* that our model cannot perfectly predict process reactions, the resulting solution hedges against this uncertainty in a very natural way.

At the same time, parallel algorithms for stochastic linear programming such as the nested decomposition algorithm [1] have brought solution times close to those acceptable in model predictive control.

The motivation for this paper comes, then, from the undesirable behavior of the LP formulation in MPC, and by the success of the stochastic approach in other fields under similar circumstances.

In Section 2, we begin with a model predictive control problem using the  $l_1$  norm, similar to the one given by Rao and Rawlings [8]. We then allow many of the fixed model parameters to be represented by discrete random vectors with finite support. The result is a multistage stochastic linear program with recourse.

In Section 3, the MSLP is broken into time stages, so that the nested decomposition algorithm for MSLPs may be applied. The resulting algorithm is given in Section 4.

## 2 Problem statement and notation

We use as a base case the following problem, using notation common in the model predictive control literature. Find  $\{x_t\}_{t=1}^T$  and  $\{u_t\}_{t=0}^T$  to

$$\begin{aligned} & \text{minimize} && \sum_{t=0}^T \|\bar{R}_t u_t\|_1 + \|\bar{Q}_t x_t\|_1 \\ & \text{subject to} && x_t = A_t x_{t-1} + B_t u_{t-1} - g_t \quad t = 1, 2, \dots, T \\ & && D_t u_t - G_t x_t \leq d_t \quad t = 0, 1, \dots, T \\ & && x_0 \text{ fixed,} \end{aligned} \tag{1}$$

where  $x_t \in \mathbb{R}^n$  represents the state of the process at time stage  $t$ , and  $u_t \in \mathbb{R}^m$  is the set of control moves to be made at time stage  $t$ . The coefficients with upper case letters are matrices, while those with lower case letters are vectors. The vectors  $g_t \in \mathbb{R}^n$  are fixed disturbances. The vectors  $d_t$  are fixed bounds. Output variables are not explicitly shown in the model. They are assumed to be linear functions of the state variable  $x_t$ , and therefore any limits on the output variables may be expressed by appropriate choices of  $D_t$ ,  $G_t$  and  $d_t$ .

To make (1) into a linear program, we introduce bounding variables  $\rho_t$  and  $\eta_t$ . The problem is then to find  $\{x_t\}_{t=1}^T$  and  $\{u_t, \rho_t, \eta_t\}_{t=0}^T$  to

$$\begin{aligned} & \text{minimize} && \sum_{t=0}^T (\mathbf{e}^\top \rho_t + \mathbf{e}^\top \eta_t) \\ & \text{subject to} && x_t = A_t x_{t-1} + B_t u_{t-1} - g_t \quad t = 1, 2, \dots, T \\ & && D_t u_t - G_t x_t \leq d_t \quad t = 0, 1, \dots, T \\ & && -\rho_t \leq \bar{R}_t u_t \leq \rho_t \quad t = 0, 1, \dots, T \\ & && -\eta_t \leq \bar{Q}_t x_t \leq \eta_t \quad t = 0, 1, \dots, T \\ & && x_0 \text{ fixed,} \end{aligned} \tag{2}$$

where  $\mathbf{e}$  represents a vector of all ones of appropriate dimension.

There are many forms that model predictive control problems may take, and the techniques of this paper are flexible enough to handle many of them. The important properties of (2) with respect to this paper are the following:

1. The objective function and all constraints are linear functions in the decision variables.
2. At least one constraint connects time periods, containing variables from two adjacent time periods. In this case, the first constraint in (2) fulfills this requirement.

Problem (2) is a multistage linear program, which can be solved in a number of ways. However, the model is deterministic. Mathematically, we are claiming that we can perfectly predict the effect of any control move on the process. It should not be surprising that this has been found [8] to often lead to overly aggressive control moves.

Following ideas from stochastic linear programming, we allow some of the problem coefficients to be random with discrete finite distributions. For  $t = 1, 2, \dots, T$ , let  $\boldsymbol{\xi}_t := (\mathbf{A}_t, \mathbf{B}_t, \mathbf{g}_t, \mathbf{D}_t, \mathbf{G}_t, \mathbf{d}_t, \bar{\mathbf{R}}_t, \bar{\mathbf{Q}}_t)$  be a random vector of appropriate dimensions, the realization of which comes at time stage  $t$ . In this paper, a boldface letter will represent a random vector, and a corresponding non-boldface letter will represent the realization of the random vector. The present is denoted by  $\xi_0$  (treating it as a realization of a random vector  $\boldsymbol{\xi}_0$ ), and  $(D_0, G_0, d_0, \bar{R}_0, \bar{Q}_0)$  are known. For  $t = 1, 2, \dots, T$  the random vector  $\boldsymbol{\xi}_t$  has a probability distribution that is conditional upon the realizations  $(\xi_0, \xi_1, \dots, \xi_{t-1})$ , discrete and finite.

The joint probability distribution of  $(\boldsymbol{\xi}_0, \boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_T)$  may therefore be represented by a scenario tree which indicates the evolving state of knowledge of the random vectors. Each time stage  $t = 0, 1, \dots, T$  has  $K(t)$  nodes in the tree, each node with its unique *subscenario*  $\sigma_t := (\xi_0, \xi_1, \dots, \xi_t)$  of realizations representing the probabilistic route to the node. Note that  $K(0) = 1$  and  $\sigma_0 = (\xi_0)$ .

The decision structure also evolves over time. Let  $z_t := (x_t, u_t, \rho_t, \eta_t)$ , and let  $\zeta_t := (z_0, z_1, \dots, z_t)$  for  $t = 0, 1, \dots, T$ . The decision  $z_t$  is made during time stage  $t$ , immediately after  $\xi_t$  is observed. Therefore  $z_t$  depends on the previous decisions  $\zeta_{t-1}$ , the previous observations  $\sigma_{t-1}$ , and the most recent observation  $\xi_t$ .

This setup, slightly unnatural in the language of mathematics, is quite natural in the real world. At each moment in time, part of the uncertain future becomes the certain present, and recourse actions may be taken in response.

We can enumerate the nodes in each stage of the scenario tree  $1, 2, \dots, K(t)$ , so that each node is uniquely identified by a  $(t, k)$  pair, where  $t$  is the time stage and  $k$  is the node number in that time stage.

The problem coefficients and variables will now be identified in this manner, so  $A(t, k)$  will be the realization of  $A_t$  for the  $k$ th node of time stage  $t$ , and so on. Similarly,  $x(t, k)$  will stand for the decision  $x_t$  made at node  $(t, k)$ , and likewise for  $u(t, k)$ ,  $\rho(t, k)$  and  $\eta(t, k)$ .

Given a node  $(t, k)$  in the scenario tree with  $t > 0$ , the function  $\mathcal{A}(t, k)$  will return the node number  $j$  such



The dual of MPC $\mathcal{I}(t, k)$  is DMPC $\mathcal{I}(t, k) :=$

$$\begin{aligned} & \text{maximize} && (d(t, k) + G(t, k)x(t, k))^\top w_1 + \\ & && \sum_{i=1}^{r(t, k)} (\alpha(t, k, i) - f(t, k, i))^\top x(t, k) w_6(i) \\ & \text{subject to} && D(t, k)^\top w_1 + \sum_{i=1}^{r(t, k)} j(t, k, i) w_6(i) = 0 \\ & && -\mathbf{e} \leq w_1 \leq 0 \\ & && -\mathbf{e} \leq w_6(i) \leq 0, \quad i = 1, 2, \dots, r(t, k). \end{aligned}$$

A feasible solution to DMPC $\mathcal{I}(t, k)$  with a strictly positive objective value shows by duality theory that MPC $(t, k)$  is infeasible for  $x(\tau, j)$  and  $u(\tau, j)$ . Expanding  $x(t, k)$  in the objective of DMPC $\mathcal{I}(t, k)$ , we have

$$\begin{aligned} & [d(t, k) + G(t, k)(-g(t, k) + A(t, k)x(\tau, j) + \\ & B(t, k)u(\tau, j))]^\top w_1 + \sum_{i=1}^{r(t, k)} [\alpha(t, k, i) - f(t, k, i)]^\top \\ & (-g(t, k) + A(t, k)x(\tau, j) + B(t, k)u(\tau, j))] w_6(i) > 0. \end{aligned}$$

Rearranging and letting

$$v := G(t, k)^\top w_1 - \sum_{i=1}^{r(t, k)} f(t, k, i) w_6(i),$$

we conclude that the condition

$$\begin{aligned} & v^\top A(t, k)x(\tau, j) + v^\top B(t, k)u(\tau, j) > \\ & -d(t, k)^\top w_1 + v^\top g(t, k) - \sum_{i=1}^{r(t, k)} \alpha(t, k, i) w_6(i) \quad (5a) \end{aligned}$$

shows infeasibility of MPC $(t, k)$ . We therefore impose a new feasibility cut on node  $(\tau, j)$  by incrementing  $r(\tau, j)$  and setting

$$f(\tau, j, r(\tau, j)) := A(t, k)^\top v, \quad (5b)$$

$$j(\tau, j, r(\tau, j)) := B(t, k)^\top v, \quad (5c)$$

and

$$\begin{aligned} & \alpha(\tau, j, r(\tau, j)) := \\ & -d(t, k)^\top w_1 + v^\top g(t, k) - \sum_{i=1}^{r(t, k)} \alpha(t, k, i) w_6(i). \quad (5d) \end{aligned}$$

Constraints of type (4f) are created to enforce the requirement that  $\theta(\tau, j)$  be bounded below by  $\mathcal{Q}_\tau(\zeta_\tau, \sigma_\tau)$ . Given  $\tau \in \{0, 1, \dots, T-1\}$ ,  $j \in \{1, 2, \dots, K(\tau)\}$ ,  $x(\tau, j)$ ,  $u(\tau, j)$  and  $\theta(\tau, j)$ , suppose that MPC $(t, k)$  is feasible for each descendant node  $(t, k) \in \{(t, k) | t = \tau + 1, k \in \mathcal{D}(\tau, j, t)\}$ . We wish to examine the quality of  $x(\tau, j)$ ,  $u(\tau, j)$  and  $\theta(\tau, j)$  with respect to optimality.

The dual of MPC $(t, k)$  may be written as

DMPC $(t, k) :=$

$$\begin{aligned} & \text{maximize} && [d(t, k) + G(t, k)x(t, k)]^\top w_1(k) + \\ & && [\bar{Q}(t, k)x(t, k)]^\top (w_4(k) - w_5(k)) + \\ & && \sum_{i=1}^{r(t, k)} [\alpha(t, k, i) - f(t, k, i)]^\top x(t, k) w_6(k, i) + \\ & && \sum_{i=1}^{s(t, k)} [\beta(t, k, i) - e(t, k, i)]^\top x(t, k) w_7(k, i) \end{aligned}$$

subject to

$$\begin{aligned} & D(t, k)^\top w_1(k) - \bar{R}(t, k)^\top (w_3(k) - w_2(k)) + \\ & \sum_{i=1}^{r(t, k)} j(t, k, i) w_6(k, i) + \sum_{i=1}^{s(t, k)} h(t, k, i) w_7(k, i) = 0 \\ & -w_2(k) - w_3(k) = \mathbf{e} \\ & -w_4(k) - w_5(k) = \mathbf{e} \\ & -\sum_{i=1}^{s(t, k)} w_7(k, i) = 1 \\ & w_1(k), w_2(k), w_3(k), w_4(k), w_5(k), w_6(k, i) \Big|_{i=1}^{r(t, k)}, \\ & w_7(k, i) \Big|_{i=1}^{s(t, k)} \leq 0. \end{aligned}$$

Suppose we find a feasible solution to each DMPC $(t, k)$  for which  $k \in \mathcal{D}(\tau, j, t)$ , such that

$$\sum_{k \in \mathcal{D}(\tau, j, t)} Y^*(t, k) p(t, k) > \theta(\tau, j), \quad (6)$$

where  $Y^*(t, k)$  is the objective value of DMPC $(t, k)$  and  $p(t, k)$  is the conditional probability of node  $(t, k)$  given  $(\tau, j)$  has been reached. This result indicates that  $\mathcal{Q}_\tau(\zeta_\tau, \sigma_\tau) > \theta(\tau, j)$ , so  $\mathcal{Q}_\tau(\zeta_\tau, \sigma_\tau)$  is not acting as a lower bound on  $\theta(\tau, j)$  as intended. Setting

$$\begin{aligned} & \nu(t, k)^\top := w_1(k)^\top G(t, k) + (w_4(k) - w_5(k))^\top \bar{Q}(t, k) - \\ & \sum_{i=1}^{r(t, k)} w_6(k, i) f(t, k, i)^\top - \sum_{i=1}^{s(t, k)} w_7(k, i) e(t, k, i)^\top, \quad (7) \end{aligned}$$

expanding (6) and rearranging, we get

$$\begin{aligned} & \sum_{k \in \mathcal{D}(\tau, j, t)} p(t, k) [\nu(t, k)^\top A(t, k)x(\tau, j) + \\ & \nu(t, k)^\top B(t, k)u(\tau, j) - \nu(t, k)^\top g(t, k) + d(t, k)^\top w_1(k) + \\ & \left. \sum_{i=1}^{r(t, k)} \alpha(t, k, i) w_6(k, i) + \sum_{i=1}^{s(t, k)} \beta(t, k, i) w_7(k, i) \right] > \theta(\tau, j). \end{aligned}$$

We add a cut of the form (4f) to node  $(\tau, j)$  by incre-

menting  $s(\tau, j)$  and setting

$$e(\tau, j, s(\tau, j)) := \sum_{k \in \mathcal{D}(\tau, j, t)} p(t, k) A(t, k)^\top \nu(t, k), \quad (8a)$$

$$h(\tau, j, s(\tau, j)) := \sum_{k \in \mathcal{D}(\tau, j, t)} p(t, k) B(t, k)^\top \nu(t, k), \quad (8b)$$

and

$$\beta(\tau, j, s(\tau, j)) := \sum_{k \in \mathcal{D}(\tau, j, t)} p(t, k) \left[ \nu(t, k)^\top g(t, k) - \right. \\ \left. d(t, k)^\top w_1(k) - \sum_{i=1}^{r(t, k)} \alpha(t, k, i) w_6(k, i) - \sum_{i=1}^{s(t, k)} \beta(t, k, i) w_7(k, i) \right]. \quad (8c)$$

We now have the pieces of a nested decomposition algorithm for solving (3), and may put them together to form a coherent algorithm.

#### 4 Nested decomposition algorithm

The nested decomposition algorithm for stochastic linear programming works on a problem that is decomposed into time stages, such as (3). Information flows iteratively up and down the scenario tree in the following ways. Decision variable values are given from a node to all of its descendants, influencing feasibility and optimality of the descendant nodes. The descendant nodes give feedback on these values to the ancestor node by means of cutting planes of type (4e) and (4f).

Because there are more than two time stages, we have flexibility with respect to how we traverse up and down the scenario tree. For example, we could begin from node (0, 1) traveling down the tree, and change direction as soon as any cut is encountered. Other choices are listed by Gassmann [4]. The one used here is *fast forward fast back*, which is the one that on average performed best in the tests conducted by Gassmann [4]. The algorithm is given in Algorithm 1, with a subroutine given in Algorithm 2.

The Solve( $\cdot$ ) routine used in Algorithms 1 and 2 behaves as follows. It takes as input a primal or dual problem (e.g. DMPCI(2,7)) and returns the optimal objective value (assuming it exists) to that problem. Also, if the problem to be solved is of the type MPC( $t, k$ ) and  $s(t, k) = 0$ , then the constraint  $\theta(t, k) = 0$  is added to the problem before solving. When solving DMPC( $t, k$ ), if  $s(t, k) = 0$ , then obviously the  $w_7$  variables are ignored. Any LP method may be used in Solve( $\cdot$ ), including a simplex method.

Warm starts are used by saving the most recent optimal basis for each problem, by node. For example, if DMPC( $t, k$ ) has been solved in a previous iteration, we may use the previous optimal DMPC( $t, k$ ) basis as an initial feasible basis for a new DMPC( $t, k$ ) problem. This is because the only things that can change in DMPC( $t, k$ ) from one iteration to the next are  $x(t, k)$ ,  $r(t, k)$ , and  $s(t, k)$ . We may set as nonbasic the  $w_6(i)$  and  $w_7(i)$  variables corresponding to the cuts added subsequent to the previous solution of DMPC( $t, k$ ).

---

#### Algorithm 1 The main algorithm

---

**Initialization:**

**for**  $t = 0$  to  $T$  **do**

**for**  $k = 1$  to  $K(t)$  **do**

$r(t, k) := 0$ ;  $s(t, k) := 0$

        solvemeagain( $t, k$ ) := yes

        basis( $t, k$ ) := empty; basisI( $t, k$ ) := empty

**end for**

**end for**

Solve( MPC(0,1) ); save basis(0,1)

solvemeagain(0,1) := no

direction := +1;  $\tau := 0$ ; cutfound := no

**while**  $0 \leq \tau \leq T - 1$  **do**

    jumpout := no;  $j := 1$

**if**  $\tau = T - 1$  **then**

        direction := 0

**end if**

**while**  $1 \leq j \leq K(\tau)$  **and** jumpout = no **do**

        jumpout = Findacut( $\tau, j, \text{direction}, \text{cutfound}$ )

$j++$

**end while**

**if**  $\tau = T - 1$  **then**

        direction := -1

**end if**

**if**  $\tau = 0$  **and** direction  $\leq 0$  **then**

        direction := +1

**if** cutfound = yes **then**

        cutfound = no

**if** solvemeagain(0,1) = yes **then**

            Solve( MPC(0,1) ); save basis(0,1)

            solvemeagain(0,1) = no

**end if**

**else**

**STOP:** optimal

**end if**

**else**

$\tau := \tau + \text{direction}$

**end if**

**end while**

---

Similarly, a basis saved from the previous solution of DMPCI( $t, k$ ) may be used as an initial feasible basis for a new DMPCI( $t, k$ ) problem.

Note that the only node for which the primal problem MPC( $t, k$ ) is solved is (0, 1). In this case, the basis

saved is the primal basis. For all other nodes the dual problem is solved, and a dual basis is saved.

---

**Algorithm 2** Subroutine Findacut

---

**Require:**  $(\tau, j, \text{direction}, \text{cutfound})$   
 $t := \tau + 1$ ; jumpout := no  
**for all**  $k \in \mathcal{D}(\tau, j)$ , **while** jumpout = no **do**  
  **if** solvemeagain( $t, k$ ) = yes **or** direction  $\geq 0$  **then**  
    calculate  $x(t, k)$  using (4g)  
     $Z^* := \text{Solve}(\text{DMPCI}(t, k))$ ; save basisI( $t, k$ )  
    **if**  $Z^* > 0$  **then**  
      calculate  $v, f, j, \alpha$  as in (5)  
       $r(\tau, j) ++$ ; add  $f, j, \alpha$  to MPC( $\tau, j$ )  
      direction :=  $-1$ ; jumpout := yes;  
      cutfound := yes; solvemeagain( $\tau, j$ ) := yes  
    **else**  
       $Y^*(t, k) := \text{Solve}(\text{DMPC}(t, k))$ ;  
      save basis( $t, k$ )  
      **if** direction  $\leq 0$  **then**  
        calculate  $\nu, e, h, \beta$  by summation as in (7)  
        and (8)  
      **else**  
        calculate  $u(t, k), \theta(t, k)$  from the optimal  
        dual solution  
      **end if**  
      solvemeagain( $t, k$ ) := no  
    **end if**  
  **end for**  
**if** jumpout = no **and** direction  $\leq 0$  **and**  $h^\top u(\tau, j) + e^\top x(\tau, j) > \beta + \theta(\tau, j)$  **then**  
   $s(\tau, j) ++$ ; add  $h, e, \beta$  to MPC( $\tau, j$ )  
  cutfound = yes; solvemeagain( $\tau, j$ ) = yes  
**end if**  
**return**(jumpout)

---

The nested decomposition algorithm has often been called “embarrassingly parallel.” Here, the outer **for** loop in Algorithm 2 may be done in parallel, because the work done inside the loop for each descendant node ( $t, k$ ) is independent of that for the other descendant nodes. The various slave machines need only communicate with the master machine, not with each other, and said communication is minimal.

## 5 Conclusion

We have taken an MPC problem which has multiple time stages, linear objective function, and linear constraints, and allowed most of the model parameters to be discrete random vectors with finite support. This turned the problem into a multistage stochastic linear program, to which we applied a nested decomposition algorithm.

The motivation for allowing the model to be stochas-

tic is to combine the speed of LP solvers with more desirable solutions that have been seen with stochastic models in other fields.

It remains to be seen whether this algorithm, even in parallel, is fast enough to be useful in the world of model predictive control, and whether the behavior of the LP solutions will be improved upon in this application. The construction of an example, and coding and testing of the algorithm should be done with the assistance of someone with more experience (than the author) in the field of model predictive control.

## References

- [1] J. R. Birge, C. J. Donahue, D. F. Holmes, and O. G. Svintsitski. A parallel implementation of the nested decomposition algorithm for multistage stochastic linear programs. *Mathematical Programming*, 75(2):327–352, 1996.
- [2] D. R. Cariño and W. T. Ziemba. Formulation of the Russell-Yasuda Kasai financial planning model. *Operations Research*, 46(4):433–449, July-August 1998.
- [3] Prashant Dave, Dennis A. Willig, Gautham K. Kudva, Joseph F. Pekny, and Francis J. Doyle. LP methods in MPC of large-scale systems: application to paper-machine CD control. *AIChE Journal*, 43(4):1016–1031, 1997.
- [4] H. I. Gassmann. MSLiP: a computer code for the multistage stochastic linear programming problem. *Mathematical Programming*, 47:407–423, 1990.
- [5] P. Klaassen, J. F. Shapiro, and D. E. Spitz. Sequential decision models for selecting currency options. Technical Report IFSRC No. 133-90, International Financial Services Research Center, Massachusetts Institute of Technology, July 1990.
- [6] A. Martel and A. Al-Nuaimi. Tactical manpower planning via programming under uncertainty. *Operations Research Quarterly*, 24:571–585, 1973.
- [7] J. L. Midler and R. D. Wollmer. Stochastic programming models for scheduling airlift operations. *Naval Research Logistics Quarterly*, 16:315–330, 1969.
- [8] Christopher V. Rao and James B. Rawlings. Linear programming and model predictive control. *Journal of Process Control*, 10:283–289, 2000.