17th European Symposium on Computer Aided Process Engineering – ESCAPE17
V. Plesu and P.S. Agachi (Editors)

Robust dynamic programming via multi-parametric programming

Nuno P. Faísca,[a] Kostas I. Kouramas,[a] Pedro M. Saraiva,[b] Berç Rustem[a] and Efstratios N. Pistikopoulos[b]

[a] *Centre for Process Systems Engineering, Imperial College London, SW7 2AZ, UK,{n.faisca, k.kouramas, br, e.pistikopoulos}@imperial.ac.uk*
[b] *GEPSI-PSE group,University of Coimbra, 3030-290 Coimbra,Portugal, pas@eq.uc.pt.*

Abstract

In this work, we present a new algorithm for solving complex multi-stage optimisation problems involving hard constraints and uncertainties, based on dynamic and multi-parametric programming. Each echelon of the dynamic programming procedure, typically employed in the context of multi-stage optimisation models, is interpreted as a robust multi-parametric optimisation problem, with the present states and future decision variables being the parameters, while the present decisions the corresponding optimisation variables. This reformulation significantly reduces the dimension of the original problem, essentially to a set of lower dimensional multi-parametric programs, which are sequentially solved. Furthermore, the use of sensitivity analysis circumvents non-convexities that naturally arise in constrained dynamic programming problems. The application of the proposed novel framework to robust constrained optimal control is highlighted.

Keywords: multiparametric programming, dynamic programming.

1. Introduction

Multi-stage decision processes have attracted considerable attention in the open literature. With many applications in engineering, economics and finances, theory and algorithms for multi-stage decision problems have been broadly presented [1,2]. A typical multi-stage optimisation problem, involving a discrete-time model and a convex stage-additive cost function, can be posed as follows [2,3]:

$$x_{k+1} = f_k(x_k,u_k),\ x_k \in \boldsymbol{X},\ u_k \in U_k,\ k \in \{0,1,\ \ldots,\ N\text{-}1\}, \tag{1a}$$

$$J(U) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k), \tag{1b}$$

where, k indexes discrete time, x_k is the state of the system at time k, $\boldsymbol{X} \subseteq \mathrm{IR}^n$, u_k denotes the optimisation (control) variable at time k, $U \equiv \{u_0,\ u_1,\ \ldots,\ u_{N-1}\}$, $U \subseteq \mathrm{IR}^m$, f_k describes the dynamic behaviour of the system and g_k is the cost occurred at time k. Based on a sequence of stage-wise optimal decisions, the system transforms from its original state, x_0, into a final state, x_N. The set of optimal decisions, $\{u_0^*,\ u_1^*,\ \ldots,\ u_{N-1}^*\}$, and the corresponding path, $\{x_1^*,\ x_2^*, \ldots, x_N^*\}$, optimise a pre-assigned cost function (1b). In other words, if the sequence of decisions is optimal the reward is maximum.

Dynamic Programming is well-documented [1] as being a powerful tool to solve this class of optimisation problems. Based on the optimality principle, the original problem disassembles into a set of problems of lower dimensionality, thereby reducing the complexity of obtaining the solution. The *value function* for a general multi-stage optimisation problem, as in (1), is given by:

$$V_k(x_k) = \min_{\mu_k,\ldots,\mu_N} [g_N(x_N) + \sum_{i=k}^{N-1} g_i(u_i, x_i)], \tag{2}$$

where $u_i = \mu_i\ (x_i) \in U_i$, and μ_i (·) is an admissible policy. Applying the optimality principle to Equation (2) results in the following recursive equation[3]:

$$V_k(x_k) = \min_{\mu_k \in U_k} [g_k(u_k, x_k) + V_{k+1}(x_{k+1})]. \tag{3}$$

From (3) we conclude that incumbent cost functions are a compound of all future cost functions, previously optimised, and the cost corresponding to the decision taken at the present time. Bellman [1] proved that this methodology solves the original problem to global optimality. The obvious advantage is that at each time step/stage the decision maker just takes one decision, provided that all future stages are optimised up to the incumbent stage.

Although dynamic programming is a well-established methodology, a number of limitations can be identified. For instance, in the linear-quadratic regulator control problem, dynamic programming procedure results in: $u_0 = K_0\ x_0;\ u_1 = K_1\ x_1;\ \ldots\ ;\ u_{N-1} = K_{N-1}\ x_{N-1}$, where the control action is set to be

admissible, $u_k \in U_k$. However, if the problem has hard constraints the complexity of the implementation significantly increases, mainly because optimisation over hard constraints directly results in non-linear decision laws \cite{raw1999}. Therefore, whereas the inclusion of future linear control laws in the incumbent cost function may not result in an increase of complexity, the inclusion of non-linear control laws in the incumbent cost function, even if convex, may require specialised global optimisation techniques for its solution.
Borrelli *et al.* [5] presented an attempt to solve the hard constrained multi-stage problem in a dynamic programming fashion. Based on multi-parametric programming theory [6] and on Bellman's optimality principle, the authors compute, for each stage, the corresponding control law, $u_k = \mu_k (x_k)$, using multi-parametric programming algorithms [6,7]. The key idea is to incorporate this conditional piecewise linear function in the cost function of the previous stage, reducing it to a function of only the incumbent stage variables, u_{k-1} and x_{k-1}. However, as the objective function at each stage is a piecewise quadratic function of $\{x_k, u_k\}$ overlapping critical regions result, and a parametric global optimisation procedure is thus required to obtain the explicit solution.
In this work, we present a novel algorithm for the solution of constrained dynamic programs which effectively avoids the need for any global optimisation procedure. The algorithm combines the principles of multi-parametric programming [6] and dynamic programming, and can readily be extended to handle uncertainty in the model data [8-10], as described next.

2. Methodology

The main steps of our approach are summarised in Figure 1 [6]. Here, we will illustrate in detail how the algorithm can be applied in the context of robust optimal control, by revisiting a popular control example problem [11]:

Algorithm
Step 1. (*j=1*) Solve the N^{th} stage of the problem, considering it as a multi-parametric optimisation problem, with parameters being the incumbent state-space, x_{N-1};
Step 2. ($j = j + 1$) Solve the $(N - j + 1)^{th}$ stage of the problem, considering it as multi-parametric optimisation problem, with parameters being the incumbent state-space, x_{N-j} and the future optimisation (control) variables, $u_{N-j+1}, \dots, u_{N-1}$;
Step 3. Compute the optimal control action for sample time j, comparing the two sets obtained in the steps before, $u_{N-j+1} = \mu_{N-j+1}(u_{N-j+2}, \dots, u_{N-1}, x_{N-j+1})$, (if $j = 2 \Rightarrow u_{N-1} = \mu_{N-1}(x_{N-1})$), and $u_{N-j} = f_{N-j}(u_{N-j+1}, \dots, u_{N-1}, x_{N-j})$, and compute, $u_{N-j} = \mu_{N-j}(x_{N-j})$;
Step 4. If j = N stop. Else go to Step 1.

Figure 1. Dynamic programming via multi-parametric programming

$$\min_U J\{U,x\} = x'_N \cdot \mathbf{P} \cdot x_N + \sum_{k=0}^{N-1} [x'_k \cdot \mathbf{Q} \cdot x_k + u'_k \cdot \mathbf{R} \cdot u_k],$$
$$\begin{aligned} s.t.\ \ & x_{k+1} = A \cdot x_k + B \cdot u_k, \\ & -1 \le u_k \le 1, \qquad k = 0,1,\ldots,N-1, \\ & -10 \le x_k \le 10, \qquad k = 0,1,\ldots,N, \end{aligned} \tag{4}$$

where, $x_k \in \mathrm{IR}^2$, $u_k \in \mathrm{IR}$,

$$N = 3; A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}; B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; P = \begin{bmatrix} 1.8588 & 1.2899 \\ 1.2899 & 6.7864 \end{bmatrix}; Q = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}; R = 1.$$

We also assume unknown but bounded uncertainty in the data of matrices A,B of the dynamic model as follows: $\{A = A + \delta_1 A; -\varepsilon_1 \mid A \mid \le \delta_1 A \le \varepsilon_1 \mid A \mid\}$ and $\{B = B + \delta_1 B; -\varepsilon_1 \mid B \mid \le \delta_1 B \le \varepsilon_1 \mid B \mid\}$. Due to the presence of uncertainty, another step is required prior to the algorithm in Figure 1.

Step 0. For the linear model and path constraints, (4), the following constraints are introduced, as suggested in [8,9], to obtain a solution immune to uncertainty:

$$x_{k+1}^{min} \le Ax_k + Bu_k \le x_{k+1}^{max},$$

$$\begin{aligned} & A \cdot x_k + \epsilon_1 \cdot |A| \cdot |x_k| + B \cdot u_k + \\ & \qquad + \epsilon_2 \cdot |B| \cdot |u_k| \le x_{k+1}^{max} + \delta \cdot \max[1, |x_{k+1}^{max}|], \\ & -A \cdot x_k + \epsilon_1 \cdot |-A| \cdot |x_k| + (-B) \cdot u_k + \\ & \qquad + \epsilon_2 \cdot |-B| \cdot |u_k| \le -x_{k+1}^{min} + \delta \cdot \max[1, |x_{k+1}^{min}|], \end{aligned}$$

or, by setting δ =0, i.e., we do not allow any constraint relaxation, the following robust optimal control formulation results:

$$\min_U J\{U,x\} = x'_N \cdot \mathbf{P} \cdot x_N + \sum_{k=0}^{N-1} [x'_k \cdot \mathbf{Q} \cdot x_k + u'_k \cdot \mathbf{R} \cdot u_k],$$
$$\begin{aligned} s.t.\ \ & x_{k+1} = A \cdot x_k + B \cdot u_k, \\ & A \cdot x_k + \epsilon_1 \cdot |A| \cdot y_k + B \cdot u_k + \epsilon_2 \cdot |B| \cdot \omega_k \le 10, \\ & -A \cdot x_k + \epsilon_1 \cdot |-A| \cdot y_k + (-B) \cdot u_k + \epsilon_2 \cdot |-B| \cdot \omega_k \le 10, \\ & -y_k \le x_k \le y_k, \\ & -\omega_k \le u_k \le \omega_k, \\ & -1 \le u_k \le 1, \qquad k = 0,1,\ldots,N-1, \\ & -10 \le x_k \le 10, \qquad k = 0,1,\ldots,N. \end{aligned}$$

We are now ready to execute the remaining steps of the algorithm in Figure 1;
Step 1. Third stage – Recast the third stage optimisation problem as a multi-parametric programming program with x_2 being the parameters:

$$\min_{u_2,\omega_2,y_2} J = x_3' \cdot \mathbf{P} \cdot x_3 + u_2' \cdot \mathbf{R} \cdot u_2,$$
$$\begin{aligned} s.t.\ & x_3 = A \cdot x_2 + B \cdot u_2, \\ & + A \cdot x_2 + \epsilon_1 \cdot |A| \cdot y_2 + B \cdot u_2 + \epsilon_2 \cdot |B| \cdot \omega_2 \leq 10, \\ & - A \cdot x_2 + \epsilon_1 \cdot |-A| \cdot y_2 + (-B) \cdot u_2 + \epsilon_2 \cdot |-B| \cdot \omega_2 \leq 10, \\ & - y_2 \leq x_2 \leq y_2, -\omega_2 \leq u_2 \leq \omega_2, -1 \leq u_2 \leq 1, -10 \leq x_3 \leq 10. \end{aligned}$$

A suitable multi-parametric programming algorithm [7] can be used to obtain its solution, resulting in the decision law: $(u_2, \omega_2, y_2) = f(x_2)$, which comprises 12 critical regions;

Step 2. Incorporating the model information, $x_{k+1} = A\, x_k + B\, u_k$, (for x_2), in each critical region. For instance, in critical region #8:

$$\begin{array}{rcl} \textbf{Critical region } \#8,\ f(x_2): & \Rightarrow & \textbf{Critical region } \#8,\ f(x_1,u_1): \\ -0.385x_2^1 - x_2^2 \leq 0.800, & & -0.385x_1^1 - 1.385x_1^2 - u_1 \leq 0.800, \\ x_2^1 + 0.980x_2^2 \leq 9.90, & & x_1^1 + 1.980x_1^2 + 0.980u_1 \leq 9.90, \\ 0.385x_2^1 + x_2^2 \leq 0, & & 0.385x_1^1 + 1.385x_1^2 + u_1 \leq 0, \\ -x_2^1 \leq 0, & & -x_1^1 - x_1^2 \leq 0, \\ \textit{Optimal decision law } u_2 = f(x_2) & \Rightarrow & \textit{Optimal decision law } u_2 = f(x_1,u_1) \\ u_2 = -0.481x_2^1 - 1.25x_2^2, & & u_2 = -0.481x_1^1 - 1.73x_1^2 - 1.25u_1, \\ \omega_2 = -0.481x_2^1 - 1.25x_2^2, & & \omega_2 = -0.481x_1^1 - 1.73x_1^2 - 1.25u_1, \\ y_2^1 = x_2^1, & & y_2^1 = x_1^1 + x_1^2, \\ y_2^2 = -x_2^2, & & y_2^2 = -x_1^2 - u_1; \end{array}$$

Step 3. Second stage – Recast the second stage optimisation problem as a multi-parametric programming problem, with x_1 and u_2 being the parameters:

$$\min_{u_1,\omega_1,y_1} J = x_3' \cdot \mathbf{P} \cdot x_3 + u_2' \cdot \mathbf{R} \cdot u_2 + x_2' \cdot \mathbf{Q} \cdot x_2 + u_1' \cdot \mathbf{R} \cdot u_1,$$
$$\begin{aligned} s.t.\ & x_2 = A \cdot x_1 + B \cdot u_1, \\ & + A \cdot x_1 + \epsilon_1 \cdot |A| \cdot y_1 + B \cdot u_1 + \epsilon_2 \cdot |B| \cdot \omega_1 \leq 10, \\ & - A \cdot x_1 + \epsilon_1 \cdot |-A| \cdot y_1 + (-B) \cdot u_1 + \epsilon_2 \cdot |-B| \cdot \omega_1 \leq 10, \\ & - y_1 \leq x_1 \leq y_1, -\omega_1 \leq u_1 \leq \omega_1, -1 \leq u_1 \leq 1, -10 \leq x_2 \leq 10. \end{aligned}$$

The solution of (14) can be obtained by multi-parametric programming, resulting in explicit expressions, $u_1 = f(x_1, u_2)$, in 22 critical regions;

Step 4. We then incorporate the future decisions, $(u_2, \omega_2, y_2) = f(x_1, x_1)$, in the current decisions, $u_1 = f(x_1, u_2)$, by which we obtain expressions: $u_1 = f(x_1)$. Note that we need to incorporate the 12 regions obtained in *Step 2* in each one of the 22 regions obtained in *Step 3*, i.e. we generate 264 critical regions. Feasibility tests are performed here [6], with which infeasible critical regions are eliminated and a compact set of regions is obtained, resulting in only 80 regions to examine further. Constraints belonging to future stages are not considered, as future constraints satisfaction is implicitly guaranteed by definition of the present map of critical regions. Hence, the use of a global optimisation procedure is not required;

Step 5. First stage – Similarly, we can obtain the final map of critical regions, i.e. all feasible solutions involving 464 critical regions. Each critical region corresponds to a different policy, however, many regions may have the same identical first-stage optimal decision, u_0. In the example above, only 20 different first-stage optimal decisions were identified(i.e. a potential reduction over 95%). The implication of this in a closed-loop robust control implementation strategy, where only the first-stage decisions are updated, is that a very significant reduction of the number of critical regions (control laws) can take place, by merging the adjacent regions with identical first-stage control actions.

3. Concluding remarks

We presented an outline of the main steps of a novel multi-parametric programming approach for the solution of general, constrained convex multi-stage problems involving uncertainty. Through a literature example of optimal control problems, we highlighted how: (i) we can reformulate the original multi-stage optimal control problem involving polytopic uncertainties into its robust equivalent, while preserving the original model structure and features, (ii) we can use recently proposed multi-parametric programming theory and algorithms [6] to efficiently address constrained dynamic programming procedures, used in the context of multi-stage optimal control formulations, (iii) we can avoid any need for global optimisation methods by carefully posing and conducting feasibility tests, based on sensitivity analysis of the obtained parametric solutions. Whilst the details of the proposed theory will be described in a subsequent publication, the work presented here clearly establishes the foundations towards a comprehensive general theory for robust optimal control.

Acknowledgements

Financial support from EPSRC (GR/T02560/01) and Marie Curie European Project PRISM (MRTN-CT-2004-512233) is gratefully acknowledged.

References

1. R. Bellman, Dynamic Programming, Mineola, 2003.
2. D. Bersekas, Dynamic Programming and Optimal Control, Bellmont, 2005.
3. T. Başar and G. Olsder, Dynamic Noncooperative Game Theory, London, 1982.
4. J. Rawlings, Proceedings of the American Control Conference 1999, 1 (1999) 662.
5. F. Borrelli, Baotić, A. Bemporad and M. Morari, Automatica 41 (2005) 1709.
6. E.N. Pistikopoulos, M.C. Georgiadis and V.Dua, Multi-parametric programming: Theory, Algorithms and Applications, Weinheim, 2006.
7. V. Dua, N. Bozinis and E.N. Pistikopoulos, Comp. & Chem. Eng. 26 (2002) 715.
8. A. Ben-Tal and A. Nemirovski, Math. Program., Ser. A 88 (2000) 411.
9. C.A. Floudas, Deterministic Global Optimization: Theory, Algorithms and Applications, New York, 2000.
10. E.N. Pistikopoulos, V. Dua, N. Bozinis, A. Bemporad and M. Morari, Comput. & Chem. Eng. 26 (2002) 175