Improving Reliability of Partition Computation in Explicit MPC with MPT Toolbox

Samo Gerkšič*

** Dept. of Systems and Control, Jozef Stefan Institute, Jamova 39, Ljubljana, Slovenia (e-mail: samo.gerksic@ijs.si).*

Abstract: The paper addresses the problem of numerical issues and degeneracies in the parametric quadratic programming (pQP) algorithm, used for computing partitions of explicit model predictive controllers (eMPC) with the Multi-Parametric Toolbox (MPT). We summarise the pQP problem setup and the basic algorithm, analyse its implementation in MPT, expose the numerical issues and suggest a series of improvements for more reliable operation, which are relevant also for other pQP solvers.

Keywords: explicit predictive control, multi-parametric quadratic programming, numerical methods.

1. INTRODUCTION

Predictive control (Qin and Badgwell, 2003; Maciejowski, 2002) is a family of advanced control algorithms based on optimisation of model-predicted future course of control signals, where constraints imposed on the process signals may be taken into account.

With the recently discovered explicit (parametric) form of predictive controllers (eMPC) (Pistikopoulos, Dua, Bozinis, Bemporad, and Morari, 2000; Bemporad, Morari, Dua, and Pistikopoulos, 2002; Pannocchia, Laachi, and Rawlings, 2006; Pistikopoulos, Georgiadis, and Dua, 2007), the computational burden of on-line optimisation is shifted to the controller design phase. In the on-line part of the control algorithm, optimisation is replaced by a precomputed parametric solution of the optimisation problem, typically given in the form of a polyhedral partition of the whole relevant multi-dimensional parameter space. With constrained linear models and quadratic performance criterion, this partition is computed by solving an optimisation problem known as (multi-)parametric quadratic program (pQP). This enables controller implementation using standard industrial control hardware, such as programmable logic controllers, embedded controllers, or even FPGA chips (Johansen, Jackson, Schreiber, and Tøndel, 2007), and the application domain is expanded to processes with fast dynamics. On the other hand, due to the parametric explosion of the computational demand and memory consumption in the off-line phase, applicability is restricted to problems of relatively small dimensions. Therefore, the expected domain of application, opposed to that of conventional MPC (Qin and Badgwell, 2003), shifts towards low-level control of univariate or small-sized multivariate processes. Due to this paradigm change, the efficiency of disturbance rejection becomes more important (Gerkšič and Pregelj, 2009).

The existing implementations of eMPC (Kvasnica, 2009; Bemporad, 2006; ParOS, 2003) stem from basic research, and require a "well posed problem" for successful eMPC partition computation. Theoretical papers on eMPC are mostly focused on the basic problem of control of states towards the origin with the assumption of perfectly measured process states, which is never satisfied in practice. Applications mostly require offset-free tracking of reference signals in the presence of disturbances, which may be provided using model augmentation (Gerkšič and Pregelj, 2009). Unfortunately, the available pQP implementations are not reliable enough for solving eMPC problems involving higher-dimension models resulting from such augmentations. Problems are also commonly encountered with long horizons and short sampling times, required for efficient disturbance rejection and output constraints handling. Under such circumstances, degeneracies (regions with more active constraints than the number of optimisation variables) and various numerical problems occur, resulting in incomplete computed partitions. Therefore, direct conversion to eMPC is not enabled even for very simple offset-free tracking MPC controllers using disturbance estimation; only specially adapted tracking implementations may be used, such as those of Bemporad (2006) or Grancharova, Johansen, and Kocijan (2004), for example. More reliable solutions are essential for wider applicability of eMPC.

Following the early publications (Pistikopoulos et al., 2000; Bemporad et al., 2002), a number of improved pQP algorithms have been published. The MPT Toolbox uses an improved method of parameter space exploration, which does not require unnecessary cuts of the parameter space (Baotić, 2002; Grieder, Borrelli, Torrisi, and Morari, 2004). Tøndel, Johansen, and Bemporad (2003a) present a modified exploration procedure that mostly does not require solving a QP when determining adjacent regions; the adjacent regions are being determined by examining which constraint changes the activity status at the polyhedron boundary (however, in certain degenerate cases the use of the QP-based procedure may still be required). An extended version of this method (Tøndel, Johansen, and Bemporad, 2003b) is suitable also for problems with non-strictly positive-definite Hessian matrices, and degeneracies are solved using projection. Spjøtvold, Kerrigan, Jones, Tøndel, and Johansen (2006) have shown that in certain cases a single facet of a region may be adjacent

to several other regions; a procedure for detection and solving of such cases was proposed. Mayne and Raković (2003) and Spjøtvold, Tøndel, and Johansen (2007) propose algorithms ensuring a non-overlapping partition by using the normal cone optimality condition. Jones and Morari (2006) show that a non-overlapping partition may be ensured by solving a related parametric linear complementarity problem (pLCP), where a uniquely defined non-overlapping partition is achieved using lexicographic perturbations (a technique used to avoid cycling conditions in some active-set QP solvers).

This paper addresses issues of degeneracies and numerical problems in the pQP algorithm used in the MPT Toolbox for computing eMPC partitions for problems with constrained linear models and a quadratic performance criterion with a strictly positive definite Hessian. After summarising the pQP problem and a framework algorithm for exploring the parameter space, we analyse the MPT implementation of the algorithm, expose the numerically problematic tasks, and suggest a set of improvements that enable more reliable computation of partitions. For illustration, a simplified example of an eMPC controller with disturbance estimation which exhibits numerical problems and degeneracy is given.

2. PARAMETRIC QUADRATIC PROGRAMMING

2.1 pQP Problem Summary

Parametric quadratic programming is defined as minimisation of a quadratic cost function $f(x, \theta)$ with respect to the optimised vector $x \in \Re^n$ subject to linear inequality constraints, where the solution is a function of the *parameters vector* $\theta \in \Re^s$ on a full-dimensional set of admissible parameters $\Theta \subset \Re^s$ (Spjøtvold et al., 2006)

$$J^*(\theta) = \min_x f(x,\theta) \quad | \quad Ax \le b + S\theta \tag{1}$$

$$f(x,\theta) = \tfrac{1}{2} x^{\mathrm{T}} H x + \theta^{\mathrm{T}} F^{\mathrm{T}} x + c^{\mathrm{T}} x \tag{2}$$

where $f(x, \theta)$ is defined by $H = H^{\mathrm{T}} \in \Re^{n \times n}$, $F \in \Re^{n \times s}$ in $c \in \Re^{n \times 1}$, and linear constraints by $A \in \Re^{q \times n}$, $b \in \Re^{q \times 1}$ in $S \in \Re^{q \times s}$, respectively. The discussion is restricted to convex pQP, where the Hessian is positive-definite, $H \ge 0$. In the practically most important subclass of strictly convex pQP with $H > 0$, a substitution of the optimisation variable $z = x + H^{-1}F^{\mathrm{T}}\theta$ may be used, which simplifies the cost function to

$$f(z,\theta) = \tfrac{1}{2} z^{\mathrm{T}} H z \tag{3}$$

also requiring a suitable conversion of linear inequality constraints in (1).

The *set of active constraints* $\mathcal{A}$ comprises indices of inequalities $i \in \{1, ..., q\}$, that for some x satisfy the equality $A_i x = b_i + S_i \theta$. Its complement is the set $\mathcal{N} = \{1, 2, ... , q\} \setminus \mathcal{A}$. The submatrices $A_{\mathcal{A}}$, $b_{\mathcal{A}}$, $S_{\mathcal{A}}$ contain rows of constraints in (1) representing equality constraints, while the submatrices $A_{\mathcal{N}}$, $b_{\mathcal{N}}$, $S_{\mathcal{N}}$ represent inequality constraints. A full-rank $A_{\mathcal{A}}$ fulfills the "linear independence constraint qualification" (LICQ) condition (Tøndel, Johansen & Bemporad, 2003a). Unfulfilled LICQ condition results in *degeneracy*.

The *optimal set of active constraints* $\mathcal{A}^*$ comprises the constraints active at the optimal solution $x^*(\theta)$. The *critical region* $\Theta_{\mathcal{A}}$ for a given $\mathcal{A}$ is a subspace of the set of admissible parameters Θ, in which $\mathcal{A}^*$ corresponds to $\mathcal{A}$.

Bemporad et al. (2002) have shown that the solution to a strictly convex pQP (1)-(2) is a partition of Θ to full-dimensional polyhedral regions, $\mathcal{R} = \{R_1, ..., R_K\}$, so that $\Theta = \bigcup_{k=1}^{K} R_k$, and the interiors of R_k are mutually non-overlapping. The value function $J^*(\theta)$ is a continuous piece-wise quadratic function of θ, and the optimizer $x^*(\theta)$ a piece-wise linear function of θ.

Different pQP algorithms determine regions R_k at a given set of active constraints $\mathcal{A}$ in different ways. When the LICQ condition holds, R_k is the *closure* of $\Theta_{\mathcal{A}}$ (so that R_k is a closed set containing all points on polyhedron facets), and the different procedures produce a similar result, possibly with small differences in numerical precision. In case of degeneracy, the methods take very different approaches, and the results differ in the efficiency of covering the whole Θ and in success with producing non-overlapping regions. Only full-dimensional regions are relevant; with some cases of degeneracies, lower-dimensional $\Theta_{\mathcal{A}}$ appear at boundaries of full-dimensional regions, however those are not included in $\mathcal{R}$. Full-dimensional degenerate regions also exist, and likely cause problems in less perfected pQP algorithms. The redundant inequalities that do not define facets of the polyhedron are removed from R_k.

2.2 Framework pQP Algorithm

We summarize the framework pQP Algorithm 1, joint to various pQP methods, from Spjøtvold et al. (2006). The algorithm begins in a feasible starting point θ, computes the starting region of the partition, and then explores *adjacent* regions until the whole admissible parameters set Θ is covered by the partition $\mathcal{R}$. A temporary set of regions $\mathcal{U}$ is used, containing the already discovered regions queued for the exploration of their adjacent regions.

Algorithm 1. Framework pQP Algorithm

1: Find such starting point $\theta \in \Theta$, so that its closure $\Theta_{\mathcal{A}}$ is a full-dimensional starting region R_1.
2: $\mathcal{R} \leftarrow R_1$, $\mathcal{U} \leftarrow R_1$.
3: **while** $\mathcal{U} \neq \{\}$ **do**
4: Choose any U from $\mathcal{U}$.
5: $\mathcal{U} \leftarrow \mathcal{U} \setminus \{U\}$.
6: **for** all facets f of region U **do**
7: Find set S of full-dimensional region adjacent to U along facet f.
8: $\mathcal{U} \leftarrow \mathcal{U} \cap (S \setminus \mathcal{R})$.
9: $\mathcal{R} \leftarrow \mathcal{R} \cup S$.
10: **end for**
11: **end while**

2.3 MPT Implementation of pQP Algorithm

The pQP algorithm used for computation of eMPC partitions for problems with constrained linear process models with quadratic performance criterion in the open-source MPT Toolbox 2.6.1 of ETH Zürich (Kvasnica, 2009) (`mpt_mpqp.m`) is described by Baotić (2002) and Grieder et al. (2004). It is applicable for strictly convex pQP problems, and internally uses the simplified cost function (3). Here we analyse some of its relevant details.

2.3.1 Determining active constraints $\mathcal{A}_k$ at point θ

This is a fundamental task used in step 7 and step 1 of Algorithm 1. The following QP needs to be solved

$$z^* = \min_z \tfrac{1}{2} z' H z \;\big|\; Az \le b + S\theta \tag{4}$$

so that one can identify indices of active constraints $\mathcal{A}_k$ in the matrix inequality of linear constraints that satisfy the equality

$$Az^* = b + S\theta \tag{5}$$

In non-degenerate and numerically non-problematic cases, $\mathcal{A}_k$ matches the set of non-zero Lagrange multipliers λ^*, determined with the solution of the QP (4).

2.3.2 Determining region R_k at active constraints set $\mathcal{A}_k$

This is a fundamental task used in steps 7 and 1 of Algorithm 1. It is started by forming the matrices $A_\mathcal{A}$, $b_\mathcal{A}$, $S_\mathcal{A}$, $A_\mathcal{N}$, $b_\mathcal{N}$, $S_\mathcal{N}$, based on $\mathcal{A}_k$. Possible duplicate rows in $A_\mathcal{A}$ are removed using the function `unique` (however, this appears to be inefficient and inappropriately placed). In non-degenerate cases, a parametric solution of x^* in λ^* is determined using the KKT conditions, as described in (Bemporad et al., 2002)

$$x^*_\mathcal{A}(\theta) = -H^{-1} A_\mathcal{A}^\mathrm{T} \lambda^*_\mathcal{A}(\theta) \tag{6}$$

$$\lambda^*_\mathcal{A}(\theta) = -(A_\mathcal{A} H^{-1} A_\mathcal{A}^\mathrm{T})^{-1}(b_\mathcal{A} + S_\mathcal{A}\theta) \tag{7}$$

then the critical region is formed

$$\Theta_\mathcal{A} = \left\{ \theta \in \Theta \;\middle|\; \begin{array}{c} A_\mathcal{N} x^*_\mathcal{A}(\theta) \le b_\mathcal{N} + S_\mathcal{N}\theta \\ \lambda^*_\mathcal{A}(\theta) \ge 0 \end{array} \right\} \tag{8}$$

and if it is full-dimensional, the region R_k is obtained as its closure and by removing redundant inequalities.

The algorithm detects a degeneracy if the number of active constraints in $\mathcal{A}_k$ is higher than n (dimension of optimisation vector x, or z). When detected, the algorithm determines the set of combinations of $\mathcal{A}_k$ elements containing n elements, and the procedure for determining non-degenerate region R from $\mathcal{A}$ is used for each of the combinations. If the obtained full-dimensional regions (sans those covering the subspace of Θ previously explored in the same degeneracy) form a convex union, they get merged, otherwise they are added to the list $\mathcal{U}$ as a set of separate regions. In the latter case, undesired region overlap may occur. The most problematic issue in degeneracy handling is the inadequate degeneracy test, which leads to inappropriate performance in case of linear dependence between rows of $A_\mathcal{A}$.

2.3.3 Determining the initial region R_1 (step 1)

The algorithm attempts sets R_1 as the unconstrained region, $\mathcal{A}_1 = \{\}$, using the procedure described in Section 2.3.2, if it is found to be full-dimensional.

Otherwise, a starting point θ_Feasible is determined by solving a linear program. Then, the corresponding active constraints $\mathcal{A}_1$ in θ_Feasible are determined using the procedure in Section 2.3.1, followed by the procedure from Section 2.3.2.

2.3.4 Adjacent region search (step 7)

Adjacent region search starts by finding θ_Border as the Chebyshev centre of the facet f, which is carried out by solving a linear program. Then, a point in the interior of the adjacent region θ_Beyond is determined by moving a short step out from θ_Border in the direction normal to f. Then, $\mathcal{A}_k$ and R_k are determined using the procedures of Sections 2.3.1 and 2.3.2, respectively. In principle, only one point in the adjacent space is tested, leaving out possible other regions of the set S, which may lead to unexplored areas within Θ (Spjøtvold et al., 2006) (though the algorithm may happen to find such unexplored areas when exploring adjacent areas of other regions). On the other hand, the algorithm may obtain a larger set S in case of degeneracy; the discovered regions are not necessarily adjacent to U in this case.

2.4 Important numerical thresholds in MPT pQP

The algorithm conducts a series of numerically sensitive operations involving predefined threshold values. Inappropriate threshold values may cause unsatisfactory performance.

1. QP (4) tolerances: MPT facilitates the use of a number of external optimisation libraries that include QP solvers, however the authors do not stress the importance of bringing QP solver thresholds in line with those in the pQP algorithm (or vice versa). The internal settings of various QP solvers vary; some optimization libraries offer several QP solvers with different settings and properties. We have observed that the solver `quadprog.m` of the Matlab Optimisation Toolbox tends to wrongly return result "infeasible" occasionally – an error not appearing in the older version of the same solver, `qp.m`. Both variants are occasionally prone to *cycling*, an infinite loop due to a failure in finding a suitable working set of active constraints in a degeneracy. The same problem occurs also with the related non-explicit MPC controllers; however it may remain hidden until the systems steers into a problematic region. In similar conditions, the QP solver of the ILOG CPLEX library did not exhibit cycling.

2. Linear constraints equality threshold (5) (`zero_tol`): determines which constraints will be deemed active. The choice is influenced by the accuracy of the QP (4) solution.

3. Parametric solution (7)-(6) accuracy: the solution may be wildly inaccurate in case of poor conditioning of the matrix $A_\mathcal{A} H^{-1} A_\mathcal{A}^\mathrm{T}$ being inverted. Such poor conditioning typically results from linear dependence among rows in $A_\mathcal{A}$.

4. The removal of *zero rows* before normalisation when computing $\Theta_{\mathcal{A}}$ (8) (thresholds `abs_tol`, `rel_tol` in `normalize.m`). From (8), the inequalities defining the hyperplanes of the critical region are obtained in the form $A_{\mathrm{CR}}\theta \le b_{\mathrm{CR}}$. The inequalities containing only zero values in the corresponding row of A_{CR} and are always satisfied, regardless of θ, and are removed at this point; so are those with the norm of A_{CR} row elements below the absolute or the relative tolerance (the latter considering the corresponding element of b_{CR}). However, we believe that using `abs_tol` at this point is not suitable and that its value may be too high considering the following thresholds, so that important inequalities may happen to get removed. The appearance of similar inequalities that are infeasible regardless of θ at this point is a sign of an error.

5. Chebishev radius threshold (`abs_tol`) in the function `polytope.m`. The lower-dimensional and *nano*-regions are being removed. Eventhough the nanoregions are part of the partition theoretically, it is recommendable to discard them in practice if they are small with respect to the required solution accuracy and especially if they cannot be determined accurately due to their small size. It should be pointed out that the removal of nanoregions with respect to the Chebishev radius is sensitive to scaling of parameters within θ. Additionally, Chebishev radius calculation errors were noticed in case of duplicate rows within (A_{CR}, b_{CR}); duplicate rows are being removed while removing redundant inequalities subsequently using the function `reduce.m`).

6. LP tolerances: internal solver tolerances apply when determining the Chebishev centre on the facet f in step 7, similarly as with QP (4) above.

7. Step size from the facet f in the normal direction in step 7 (`step_size`). With a long step size, thin regions may remain undetected.

2.5 Suggested improvements of MPT pQP algorithm

In order to achieve more reliable operation of the pQP algorithm, in addition to careful adjustment of the threshold values we have also used a number of modifications to the MPT implementation of the algorithm.

Improved degeneracy detection: in Section *2.3.1* we have replaced simple counting of detected active constraints in $\mathcal{A}$ with an LICQ test. The number of active constraints is determined as the rank of $A_{\mathcal{A}}$, calculated using singular-value decomposition (with a lowest singular value threshold `rank_tol`). Linear dependence among rows of $A_{\mathcal{A}}$ may also be detected using the Matlab function `rref`, however this was found to be less reliable numerically. Then, the set of combinations of $\mathcal{A}_k$ elements containing rank($\mathcal{A}_k$) elements is determined, and the procedure for determining non-degenerate region R from $\mathcal{A}$ is used for each of the combinations (starting again with the rank test and discarding combinations with lower rank). However, the discovered regions may overlap; one may choose to discard regions where overlap is detected, but that may lead to incomplete coverage of Θ.

Additionally, the conditioning of the matrix $A_{\mathcal{A}} H^{-1} A_{\mathcal{A}}^{\mathrm{T}}$ may be tested before inversion using the function `rcond` (threshold `rel_tol*10^-4`).

We have decreased the absolute threshold at removal of "zero" rows before normalisation of the critical region (8) to `abs_tol*10^-4`, and an error is reported in case of an always infeasible row.

In order to avoid the Chebishev radius calculation error in the function `polytope.m` after normalisation, we have implemented removal of duplicate rows immediately following the normalisation (threshold `abs_tol*10^-3` for the quadratic norm of row coefficients).

3. eMPC EXAMPLE

The example is derived from a practical offset-free tracking eMPC controller based on the CFTOC problem, with offset-free tracking based on disturbance estimation (Gerkšič and Pregelj, 2009). However, the setup is simplified to an artificial three-dimensional zero-reference case already exhibiting degeneracy and numerical issues. The matrices of the basic state-space process model are

$$A = \begin{bmatrix} 0.39343641 & 0 \\ 0 & 0.96342157 \end{bmatrix},$$
$$B = \begin{bmatrix} -0.023879771 \\ -0.009434897 \end{bmatrix}, \qquad (9)$$
$$C = \begin{bmatrix} -0.26172475 & 1 \end{bmatrix}, \quad D = \begin{bmatrix} 0 \end{bmatrix}$$

Disturbance augmentation is applied, with the aim of estimation of an integrating disturbance state d using a Kalman filter

$$x_a = \begin{bmatrix} x \\ d \end{bmatrix}, \quad A_a = \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix}, \quad B_a = \begin{bmatrix} B \\ 0 \end{bmatrix}, \qquad (10)$$
$$C_a = \begin{bmatrix} C & I \end{bmatrix}, \quad D_a = D$$

The MPC controller is based on the CFTOC output-cost 2-norm problem, set up using the MPT YALMIP method with some extensions, with the following parameters: predictive horizon N = 27, control horizon N_u = 3 with control move blocking by 3 samples (starting offsets: 0, 3, 6), output cost Q_y = 1, control cost R = 10^{-2}, constraints: $0 < u < 100$ (hard), $-200 < y < 1.5$ (soft, with cost S_y = 10, using a single slack variable for the ∞-norm of all y constraints; y constraints placed sparsely to each third sample - specifically at offsets: 0, 2, 5, 8, ..., 23, 26), parameter space Θ bounds (`Pbnd`): $[-50\ -50\ -50]^T < x_{a,0} < [50\ 50\ 50]^T$. pQP parameters: solver ILOG CPLEX 10, `abs_tol` = 10^{-7}, `rel_tol` = 10^{-6}, `zero_tol` = 10^{-10}, `step_size` = `abs_tol`, `rank_tol` = 10^{-5}. Appendix A contains the pQP matrices derived by MPT YALMIP.

The pQP algorithm of MPT 2.6.1 produces a controller partition with 85 regions shown in Fig. 1 (top). During the computations there are several warnings regarding the conditioning of the computation. The inspection shows that

some points of Θ are not covered by the computed regions, as exposed on the magnified 2D cross-section in Fig. 2 (top). The result is highly dependent on the parameters of the pQP algorithm and the solvers, however reliable partitioning appears to be unreachable. The partition computed with the modified pQP algorithm shown in Fig. 1 (bottom) contains 86 regions. While the difference is hardly visible in Fig. 1, Fig. 2 (bottom) shows that the problematic degeneracy area containing 5 regions is fully covered.

The computational problems of the example persist also when the MPC horizons or model dynamics are slightly modified. The conditioning of the computation is adversely affected by the single-slack-variable technique for soft y constraints; without it, reliable partitioning to 241 regions is achieved with both versions of the algorithm.

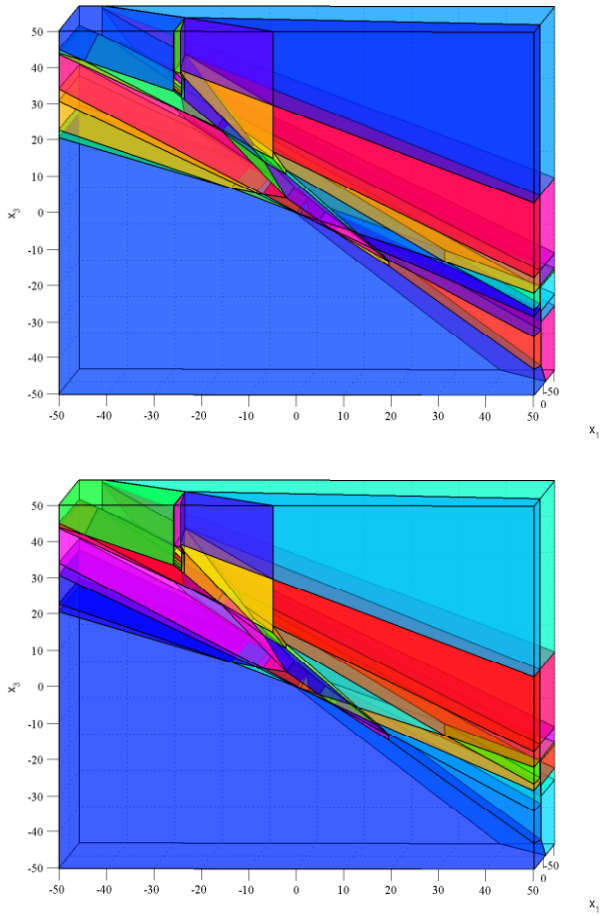

Fig. 1. Example - state partition. Top: MPT pQP; bottom: modified pQP.

4. CONCLUSION

MPT (Kvasnica, 2009) in HT (Bemporad, 2006) neither allow posing of the typical offset-free tracking eMPC problem in a traditional MPC setup using the disturbance estimation concept, nor are capable of reliable computation of the resulting controller partitions. Our results show that reliable computation of such partitions is possible by careful selection of numerical thresholds and certain improvements of the algorithms.

Although degeneracy handling has been improved considerably, the fundamental problem of possible overlapping of regions with this algorithm remains and may be overcome by using alternative pQP and pLCP algorithms. It should be pointed out that some of the suggested improvements are directly applicable to the alternative algorithms, while others have counterparts suited to the different computation methods used therein. Our preliminary results show that the algorithms produce very similar results in non-problematic regions, however there are fundamental differences in degenerate or numerically troublesome regions.

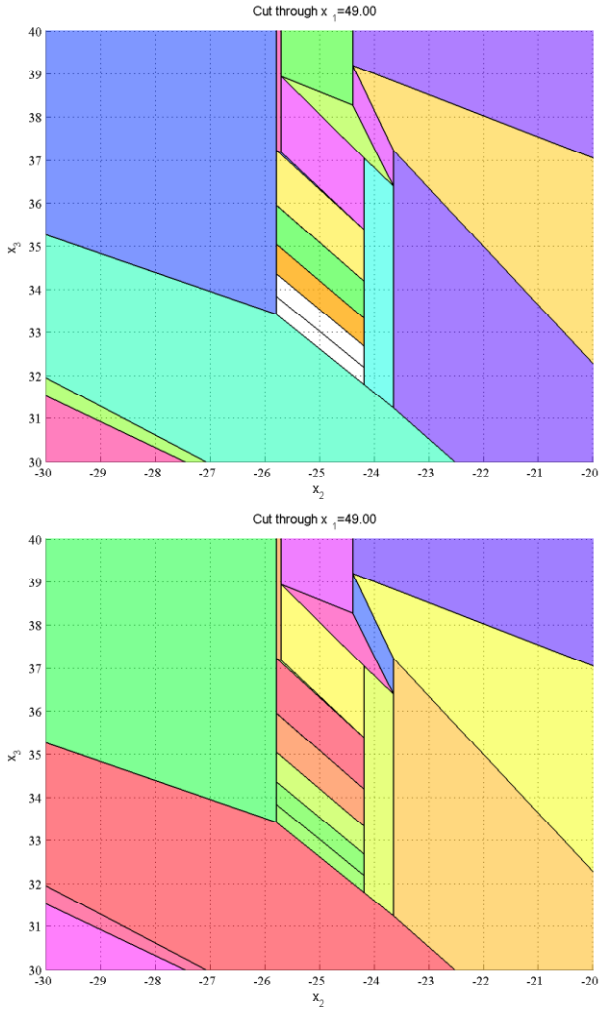

Fig. 2. Example - projected state partition. Top: MPT pQP; bottom: modified pQP.

ACKNOWLEDGEMENT

The financial support of the Slovenian Research Agency (L2-2342) is gratefully acknowledged.

REFERENCES

Baotić, M. (2002). An efficient algorithm for multi-parametric quadratic programming. Technical Report AUT02-05. ETH Zürich, Institut für Automatik, 2002.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, 38, 3–2002.

Bemporad, A. (2006). Hybrid toolbox for real-time applications, user's guide. University of Siena, Technical report.

Gerkšič, S. and Pregelj, B. (2009). Disturbance rejection tuning of a tracking multi-parametric predictive controller. *Proc. IEEE ICIT 2009*, Gippsland, 65-70.

Grancharova, A., Johansen, T. A., and Kocijan, J. (2004). Explicit model predictive control of gas-liquid separation plant via orthogonal search tree partitioning. *Computers and Chemical Engineering*, 28, 2481–2491.

Grieder, P., Borrelli, F., Torrisi, F., and Morari, M. (2004). Computation of the constrained infinite time linear quadratic regulator. *Automatica*, 40, 701–708.

Johansen, T. A., Jackson, W., Schreiber, R., and Tøndel, P. (2007). Hardware synthesis of explicit model predictive controllers. *IEEE Transactions on Control Systems Technology*, 15, 191–197.

Jones, C. N. and Morari, M. (2006). Multiparametric linear complementarity problems. *Proc. 45th IEEE Conference on Decision and Control*, 2006.

Kvasnica, M. (2009). Real-time model predictive control via multi-parametric programming. Saarbrücken: VDM verlag. Online: http://control.ee.ethz.ch/mpt/

Maciejowski, J. M. (2002). *Predictive Control with Constraints*, Harlow UK: Prentice Hall.

Mayne, D. Q. and Raković, S. V. (2003). Optimal Control of Constrained Piecewise Affine Discrete-Time Systems. *Journal of Computational Optimization and Applications*, 25, 167–191.

Pannocchia, G., Laachi, N., and Rawlings, J. B. (2006). A candidate to replace pid control: Siso-constrained lq control. *AIChE Journal*, 51, 1178–1189.

ParOS Parametric Optimisation Solutions Ltd., (2003). POP, Parametric optimization software.

Pistikopoulos, E. N., Georgiadis, M. C., and Dua, V. Eds. (2007). *Multi-Parametric Model-Based Control*. Weinheim: Wiley-VCH.

Pistikopoulos, E. N., Dua, V., Bozinis, N. A., Bemporad, A., and Morari, M. (2000). On-line optimization via off-line parametric optimization tools. *Computers and Chemical Engineering*, 24, 183–188.

Qin, S. J. and Badgwell, T. A. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11, 733–764.

Spjøtvold, J., Kerrigan, E. C., Jones, C. N., Tøndel, P., and Johansen, T. A. (2006). On the facet-to-facet property of solutions to convex parametric quadratic programs. Automatica, 42, 2209–2214.

Spjøtvold, J., Tøndel, P., and Johansen, T. A. (2007). A continuous selection and unique polyhedral representation of solutions to convex multiparametric quadratic programs. *J. Optimization Theory and Applications*, 134, 177–189.

Tøndel, P., Johansen, T. A., and Bemporad, A. (2003a). An algorithm for multiparametric quadratic programming and explicit MPC solutions. *Automatica*, 39, 489–497.

Tøndel, P., Johansen, T. A., and Bemporad, A. (2003b). Further results on multi-parametric quadratic programming. *Proc. 42th IEEE CDC*, Hawaii, 3173–3178.

APPENDIX A. EXAMPLE – pQP MATRICES

$$H = \begin{bmatrix} 0.6698 & 0.0463 & 0.0415 & 0 \\ 0.0463 & 0.0751 & 0.0145 & 0 \\ 0.0415 & 0.0145 & 0.0760 & 0 \\ 0 & 0 & 0 & 20 \end{bmatrix}$$

$$F = \begin{bmatrix} 0.0001 & -1.3601 & -2.7799 \\ 0.0015 & -0.4774 & -0.8037 \\ 0.0249 & -0.5623 & -0.8759 \\ 0 & 0 & 0 \end{bmatrix}, \quad c = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0.0852 & 0.0088 & 0.0079 & 0.6804 \\ 0.0748 & 0.0098 & 0.0087 & 0.6753 \\ 0.0632 & 0.0108 & 0.0097 & 0.6690 \\ 0.0505 & 0.0120 & 0.0107 & 0.6611 \\ 0.0366 & 0.0132 & 0.0118 & 0.6515 \\ 0.0215 & 0.0144 & 0.0130 & 0.6398 \\ 0.0061 & 0.0149 & 0.0141 & 0.6260 \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0.0060 & 0.0145 & 0.6098 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0.0058 & 0.5910 \\ 0 & 0 & 0 & 0.5709 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -5.22 \cdot 10^{-12} & 0.2582 & 0.6804 \\ -8.50 \cdot 10^{-11} & 0.2866 & 0.6753 \\ -1.383 \cdot 10^{-9} & 0.3175 & 0.6690 \\ -2.244 \cdot 10^{-8} & 0.3509 & 0.6611 \\ -3.631 \cdot 10^{-7} & 0.3867 & 0.6515 \\ -5.855 \cdot 10^{-6} & 0.4247 & 0.6398 \\ -9.405 \cdot 10^{-5} & 0.4646 & 0.6260 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -1.504 \cdot 10^{-3} & 0.5061 & 0.6098 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -0.02394 & 0.5485 & 0.5910 \\ -0.14941 & 0.5709 & 0.5709 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ -100 \\ -1.0206 \\ -1.0130 \\ -1.0034 \\ -0.9916 \\ -0.9772 \\ -0.9598 \\ -0.9390 \\ 0 \\ -100 \\ -0.9146 \\ 0 \\ -100 \\ -0.8865 \\ -0.8563 \\ 0 \\ -1000 \\ -50 \\ -50 \\ -50 \\ -50 \\ -50 \\ -50 \end{bmatrix}$$