A comparison of the computational efficiency of multi-parametric predictive control using generalised function parameterisations

B. Khan *, J.A. Rossiter *

* Automatic Control and Systems Eng., University of Sheffield, UK
e-mail: b.khan@sheffield.ac.uk, j.a.rossiter@sheffield.ac.uk

Abstract: This paper considers the computational efficiency of using generalised function parameterisations for multi-parametric quadratic programming (mp-QP) solutions to MPC. Earlier work demonstrated the potential of Laguerre parameterisations for improving computational efficiency in that the parametric solutions either required fewer regions and/or gave larger volumes. This paper considers the potential of extending this concept to more general function parameterisations. Specifically the aim is to consider to what extent different function parameterisations affect the parametric solution complexity and feasible volumes. Extensive simulation results which suggest there are indeed benefits from using more general parameterisations than Laguerre.

Keywords: Predictive control, Multi-parametric quadratic programming, Feasibility, Generalised function parametrisations.

1. INTRODUCTION

Model Based Predictive Control (MPC) or receding horizon control (RHC) or moving horizon or predictive control (Mayne et al., 2000; Rossiter, 2003; Camacho and Bordons, 2003), are general names for different computer control algorithms that use past information of the inputs and outputs and a mathematical model of the plant to optimise its predicted future behaviour. MPC is well established and widely used both in industry and control research community and has reached a high degree of maturity in its linear variant. Currently research is focused on stochastic, nonlinear and robustness issues as well as fast optimisation or related computational aspects. All available algorithms have to achieve a trade-off between feasibility, optimality and inexpensive optimisation. This paper explores the computational efficiency or benefits of deploying a generalised function parameterisation within an optimal predictive control algorithm (Scokaert and Rawlings, 1998; Kouvaritakis et al., 1998).

The success of earlier industrial heuristic MPC algorithms motivated research community to develop several algorithms with improved performance and feasible region. There are several successful theoretical approaches but few of them are exploited commercially for real-time implementation. One important issue for real-time implementation is to solve an optimisation problem within time determined by the sampling instant of the application and therefore computational efficiency of an algorithm becomes critical. A trade-off has to be made between performance, feasibility and the computational burden when choosing from the currently available algorithms. Due to the computationally expensive on-line optimisation which is required, there has been some limitation to which processes predictive control (MPC) has been used on. However, explicit solutions to the constrained MPC problem formulation (Bemporad et al., 2002) significantly increase the potential application areas for MPC. Explicit solutions to MPC problems are not intended to replace traditional implicit MPC, but rather to extend its area of applicability. MPC functionality can, with this, be applied to applications with sampling rates in the µ-sec range, using low cost embedded hardware (Canale et al., 2009). Software complexity and reliability is also improved, allowing the approach to be used on safety-critical applications.

The basic idea of the explicit solution is to solve, off-line, all possible QP problems that can arise on-line. Within certain regions, the optimum predicted input trajectory has a known affine dependence on the state; mp-QP finds all possible active sets and the associated regions and control trajectories. The online optimisation can then replaced by set membership tests; if the state is inside a particular region, the control law is made of the associated control trajectory. However, although mp-QP is transparent, it may not reduce either coding complexity or computational effort as the number of computed regions, and hence, data storage, may grow exponentially in the prediction horizon (Bemporad et al., 2002). Thus mp-QP could be unsuitable for large dimensional problems, or indeed any problem requiring a large number of regions.

The aim of this paper is to reduce the number of regions using generalised function parameterisations and therefore reduce the online computational burden (as this correlates to the number of on-line set-membership tests). Little has yet to appear in the literature which gives a significant reductions in complexity. In (Borrelli et al., 2001) the
authors reduce data storage requirements by using an evaluation of a value function for the set-membership test, but the number of regions is not reduced. (Tondel et al., 2003) introduced an efficient binary search tree, but the off-line computation of this can be prohibitive for complex controller partitions and the storage requirement may even increase. Other groups reduce the number of regions by allowing some suboptimality (Bemporad and Filippi, 2001; Grieder et al., 2003) either in the performance index or the terminal region, although preliminary results are as yet unconvincing. Another alternative is to specify regions as hypercubes (Johansen and Grancharova, 2003) to allow for efficient on-line search algorithms; however, as the structure of the controller is user-defined, it may not cover the entire controllable set. (Rossiter and Grieder, 2005) interpolates two different laws achieving a large decrease in the number of regions but determine to the performance.

This paper develops a recent contribution (Valencia-Palomo and Rossiter, 2010) to this problem which used Laguerre MPC by proposing an alternative parameterisation of the degrees of freedom (d.o.f.) in order to reduce the necessary on-line computations of optimal MPC. Specifically, the aim here is to extend this to generalised parameterisation to consider how one can reduce the data storage requirements and the online implementation time for the associated mp-QP solution. The proposed procedure is based on Kautz and generalised function parameterisations (Khan and Rossiter, 2011a, b) of the d.o.f. which provide improved feasibility and performance for the closed-loop system; we will henceforth refer to these procedures as multi-parametric KOMPC (mpKOMPC) and multi-parametric GOMPC (mpGOMPC) respectively. For illustration, a comparison between a standard multi-parametric OMPC (mpOMPC) controller and mpGOMPC controller is shown in Fig. 1 where clearly the mpGOMPC parametric solution has both fewer regions and a larger feasible volume.

Section 2 will give the necessary background about problem formulation, predictive control, Laguerre optimal predictive control (LOMPC) and Kautz optimal predictive control (KOMPC). Section 3 presents generalised function parameterisation to optimal predictive control and proposed multi-parametric QP based algorithm for GOMPC (mpGOMPC). Extensive simulation results with simulation setup are discussed in Section 4, showing the efficacy of free control moves can be used (Rossiter, 2003). For these cases, (3) is implemented (Rossiter, 2003; Scokaert and Rawlings, 1998) by imposing that the state \( x_{nk} \) must contained in an invariant set \( \chi_c \). This invariant set is also known as maximum admissible set (MAS) which satisfies all polytopic constraints with recursive use of the terminal control law \( u_k = -Kx_k \), \( x_k \in \chi_0 \). The MAS is defined as

\[
\chi_0 = \{ x_0 \in \mathbb{R}^{n_x} \mid \underline{x} \leq x_0 \leq \overline{x} \},
\]

where \( \underline{x} = -Kx_k \leq \overline{x}, x_{k+1} = A x_k + B u_k, \forall k \geq 0 \). In compact form defined as \( \chi_0 = \{ x_k : M x_k \leq b \} \) for suitable \( M \) and \( b \).

For convenience, the degree of freedom can be reformulated in terms of a new variable \( c_k \) (Rossiter, 2003; Scokaert and Rawlings, 1998)

\[
u_k = -Kx_k + c_k, \quad k = 0, ..., n_c - 1,
\]

and hence the equivalent optimisation to (3) is

\[
\min C^T SC \quad s.t. \quad M x_k + N C \leq b;
\]

where \( C = [c_k^T, ..., c_{k+n_c}^T]^T \). Details of how to compute positive definite matrix \( S \), matrices \( N, M \) and vector \( b \) are omitted as by now well known in the literature (Mayne et al., 2000; Rossiter, 2003; Gilbert and Tan, 1991).

The maximal control admissible set (MCAS) \( \chi_c \), the feasible set for optimal control problem in (6) that satisfies all polytopic constraints, is defined as

\[
\chi_c = \{ x_k : \exists C, M x_k + N C \leq b \}.
\]
Algorithm 2.1. OMPC
\[
c^*_k = \arg \min_{c_k} J_c \quad \text{s.t.} \quad Mx_k + Nc_k \leq b;
\]
Implement \(u_k = -Kx_k + e^T_1c^*_k\).

Where \(e^T_1 = [I, \ldots] \rho\) (16) with \(c_{k+i} = G(i)^T \rho\) (Khan and Rossiter, 2011b). The MCAS is calculated in a similar manner to Algorithm 2.1. OMPC

\[
\gamma^*_k = \arg \min_{\gamma_k} J_{KOMPC} \quad \text{s.t.} \quad Mx_k + NH_K \gamma_k \leq b
\]

(13)

Define \(c^*_k = K(0)^T \gamma^*_k\) and implement the control law
\[
u_k = -Kx_k + e^T_1c^*_k
\]

The details of how to define \(J_{KOMPC}\), \(H_K\) are omitted (Khan and Rossiter, 2011a).

3. GENERALISED FUNCTION PARAMETERISATION IN OMPC USING MP-QP

Laguerre and Kautz are 1st and 2nd order parameterisations and thus it is logical to consider whether higher order or generalised parameterisation techniques further improve the feasible region while maintaining performance. Generalised higher order orthonormal functions with ‘n’ pole (i.e. ‘a_1, \ldots, a_n’) network define multiple time scales for the input perturbations to improve the feasibility without deploying large numbers of d.o.f. (\(n_c\)) (Khan and Rossiter, 2011b). Laguerre and Kautz functions are special cases of generalised functions. The generalised parameterisation (Khan and Rossiter, 2011b) is defined using a higher order network such as
\[
G_i(z) = G_{i-1}(z) \frac{(z^{-1} - a_1) \cdots (z^{-1} - a_n)}{(1 - a_1 z^{-1}) \cdots (1 - a_n z^{-1})}; \quad 0 \leq a_k < 1, \quad k = 1, \ldots, n
\]

with \(G_1(z) = \sqrt{1 - a^2} \cdots (1 - a^2_1)(1 - a^2_2)\). In discrete state space form \(G_{i+1} = A_G G_i\).

3.1 Generalised OMPC or GOMPC

OMPC is, by design, based around the unconstrained optimal sequence. This basic concept is preserved and thus alternative parameterisation is used, not to model the input trajectories directly but rather the perturbations \(c_k\) around the unconstrained optimal. Hence the input prediction perturbation using a generalised function (Khan and Rossiter, 2011b) is given by

\[
C = \begin{pmatrix} c_k \\ c_{k+1} \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} G(0)^T \\ G(1)^T \\ \vdots \\ G(n)^T \end{pmatrix} \rho = H_G \rho
\]

(15)

where \(\rho\) is the \(n_G\) dimension decision variable when using the first \(n_G\) column of \(H_G\). The difference between Laguerre and Kautz function parameterisation is the definition of the perturbation signals i.e. \(c_{k+i} = K(i)^T \gamma_i\). Algorithm 2.3. The KOMPC algorithm is summarized as

\[
\gamma^*_k = \arg \min_{\gamma_k} J_{KOMPC} \quad \text{s.t.} \quad Mx_k + NH_K \gamma_k \leq b
\]

(13)

Define \(c^*_k = K(0)^T \gamma^*_k\) and implement the control law
\[
u_k = -Kx_k + e^T_1c^*_k
\]

The details of how to define \(J_{KOMPC}\), \(H_K\) are omitted (Khan and Rossiter, 2011a).
2.2. The main differences in the calculation are the use of transformation matrix $H_G$ instead of $H_L$.

3.2 Generalised OMPC solved by mp-QP or mpGOMPC

The solution of the optimisation problem using mp-QP can be applied to LOMPC, KOMPC and GOMPC as they all take the form of standard quadratic programmes; this section summarises the key points for completeness.

If the optimisation problem is solved parametrically as an explicit function of the initial conditions $x_0$, the optimal feedback law $u = f(x_0)$ takes a form of a lookup table. The on-line optimisation of such table then reduces to a simple set membership test, also known as the point location problem. Here, the table has to be searched through and the element which contains the current state measurement has to be found. The algorithm based on mp-QP for GOMPC is presented in Algorithm 3.1.

Algorithm 3.1. Off-line tasks:

1. Solve parametrically the following QP.

   \[
   \rho = \text{arg} \min_{\rho} \ J_G \\
   \text{s.t.} \quad Mx_k + NH_G \rho_k \leq b 
   \]

2. Store the optimal predicted input trajectories $\rho$ (which implies $C_G = G^T \rho$) and associated regions.

On-line tasks:

1. Find the corresponding solution ($C_G$) of the optimisation problem (17) associated to the current state $x_k$.
2. Implement the control law $u_k = -Kx_k + e_j^T C_G$, where $e_j$ is the $j$th standard basis vector.

The next section presents extensive simulations in order to show that the number of regions obtained with mpGOMPC is typically lower, for the same volume, than from mpOMPC, mpLOMPC and mpKOMPC.

Remark 3.1. It is straightforward to show, with conventional arguments, that all algorithms (i.e. LOMPC, KOMPC, GOMPC) give guaranteed recursive feasibility and stability in the nominal case and offset free tracking whenever the set point is feasible.

Remark 3.2. Note that the procedures in (Borrelli et al., 2001; Tondel et al., 2003) may be used in combination with mpGOMPC to obtain even greater reductions in complexity.

4. NUMERICAL EXAMPLES

The purpose of this section is to compute the mp-QP solution for the alternative parameterisations algorithms and then compare them to the standard OMPC algorithm. Our prime interest is to compare two aspects: (i) the size of the feasible regions; (ii) the complexity of the different algorithm solutions (in essence the number of regions) so that some comments can be made about the potential of the mp-QP solution to GOMPC.

4.1 Simulation Setup

The optimal predictive controller with $n_c = 3$ is used as a basis for comparisons. All algorithms (OMPC, LOMPC, KOMPC and GOMPC) provide stability and feasibility properties but the volume of the maximum controller admissible control admissible set (MCAS) for each of them varies. Hence, it is necessary to compare both the complexity of the algorithms and the volumes of the associated MCAS. The comparisons are based on 10 random systems with $x \in \mathbb{R}^2$, $x \in \mathbb{R}^3$ and $x \in \mathbb{R}^4$ (total of 30 systems). The inputs and states for all systems were constrained to $-1 \leq u \leq 1$ and $-10 \leq x_k \leq 10$, ($k = 2, 3, 4$) with performance objective weighting matrices $R = I$, $R = 0.1I$, $R = 10I$ and $Q = \text{diag}(1, 0, \ldots, 0)$. For simplicity the Laguerre, Kautz and generalised functions were based on $a = 0.5$, $(a_1, a_2) = (0.55, 0.57)$ and $(a_1, \ldots, a_3) = (0.58, 0.6, 0.65)$.

4.2 Volume comparisons

The feasible volume is estimated using a large number of equi-spaced directions in the state space. For each direction, the distance from the origin to the boundary of the MCAS is determined; the larger the distance, better the feasibility. Finally these distances are normalised against the distance obtained with OMPC with $n_c = 20$, we realise this is somewhat arbitrary but it seems a pragmatic limit for the global feasible region with sensible sampling and dynamics.

Feasible volumes are compared in table 1, figure 3, 5 and 7 for 30 random systems. The x-axis serves as index for the dynamic systems (10 for 2, 3 and 4 states, respectively) and the y-axis indicates the normalised feasible volume for each system. Alternative parameterisations have noticeably better feasibility than OMPC and it is clear that GOMPC has a larger MCAS than OMPC, LOMPC and KOMPC. GOMPC gets within more than 90% of the global MCAS (OMPC with $n_c = 20$) with just a 3rd order function dynamic parameterisation, and significantly deploying only $n_c = 3$!

Table 1. Complexity Vs Feasible Volume

<table>
<thead>
<tr>
<th></th>
<th>Number of regions</th>
<th>Feasible volume</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R=0.1</td>
<td>R=1</td>
</tr>
<tr>
<td>OMPC</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>LOMPC</td>
<td>21</td>
<td>16</td>
</tr>
<tr>
<td>KOMPC</td>
<td>21</td>
<td>16</td>
</tr>
<tr>
<td>GOMPC</td>
<td>22</td>
<td>17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Number of regions</th>
<th>Feasible volume</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R=0.1</td>
<td>R=1</td>
</tr>
<tr>
<td>OMPC</td>
<td>22</td>
<td>18</td>
</tr>
<tr>
<td>LOMPC</td>
<td>24</td>
<td>12</td>
</tr>
<tr>
<td>KOMPC</td>
<td>28</td>
<td>12</td>
</tr>
<tr>
<td>GOMPC</td>
<td>31</td>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Number of regions</th>
<th>Feasible volume</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R=0.1</td>
<td>R=1</td>
</tr>
<tr>
<td>OMPC</td>
<td>176</td>
<td>108</td>
</tr>
<tr>
<td>LOMPC</td>
<td>279</td>
<td>169</td>
</tr>
<tr>
<td>KOMPC</td>
<td>301</td>
<td>187</td>
</tr>
<tr>
<td>GOMPC</td>
<td>154</td>
<td>104</td>
</tr>
</tbody>
</table>
4.3 Complexity comparisons

The computational complexity is compared by finding all possible active sets and the associated regions and control trajectories using solution of mp-QP. The online computational load for the set membership test is proportional to the total number of regions.

Figure 2, 4, 6 and table 1 give a comparison of the computational complexity of mpOMPC, mpLOMPC, mpKOMPC and mpGOMPC versus 30 random systems. It is noted that the average storage requirements of mpOMPC is less but feasibility is severely limited. The alternative function parameterisations improved the feasible volume but typically increase the storage requirements of mp-QP partitions. However this is no generic pattern.

It can be noticed that:
- For 2-dimensional random systems the average storage requirement is the same for all the parameterisations for OMPC. The average storage requirement of mpGOMPC is $3 - 8$ regions more than for mpOMPC. It gives $6 - 12\%$ increase in average feasible volume.
- For 3-dimensional random systems the average storage requirement of mpKOMPC and mpLOMPC are below $1 - 3$ regions than mpGOMPC. It gives $7 - 13\%$ increase in average feasible volume with $5 - 9$ regions more than the standard mpOMPC.
- For 4-dimensional random systems the average storage requirement of mpGOMPC is below $1 - 22$ regions than the standard mpOMPC and gives $10 - 23\%$ increase in average feasible volume.
- The average storage requirement for 4th and higher order dimensional system is reduced with improved average feasible region using mpGOMPC algorithm as compared with the standard mpOMPC.
- The improvement in the number of regions does not imply too much performance loss (Bemporad and Filippi, 2001; Grieder et al., 2003), since an alternative parameterisation algorithm is an optimal algorithm.
- An interesting observation on comparing the storage requirement is that for 2nd and 3rd order systems the best choice after mpGOMPC is mpLOMPC and mpKOMPC respectively. These results indicate the possible choice for the order of parameterisation i.e. select an $n$-th order parameterisation function dynamics for $n + 1$-order system.

Remark 2. These results are based on default choices for the parameters in GOMPC. Further improvements are possible by tailoring these parameters to the context.

5. CONCLUSION AND FUTURE WORK

The paper has shown the potential benefits of generalised functions as an alternative parameterisation for improving the computational complexity in conventional MPC algorithms with a fixed number of d.o.f. It is shown through extensive simulations that computational benefits may be achieved using explicit solutions of a GOMPC algorithm. A pragmatic choice for selection of order is demonstrated using different dimensional random examples.

Future work will investigate in parallel issues such as: which alternative parameterisation is best for a particular problem and what choice of parameter(s) within that parameterisation for any given problem. The main idea is to
propose an optimisation problem based on pole locations, order of parameterisation, performance, feasibility and computational complexity. The solution of optimisation will compute the best parameterisation dynamics.

REFERENCES
Khan, B. and Rossiter, J.A. (2011a). Exploiting Kautz functions to improve feasibility in MPC. *Proc. 18th IFAC World Congress*.