Hot-start efficiency of Quadratic Programming algorithms for fast Model Predictive Control: A comparison via an Adaptive Optics case study

Mikhail Konnik\textsuperscript{1} and José De Doná\textsuperscript{2}

\textbf{Abstract}—Model Predictive Control (MPC) with fast sampling rates can be extremely demanding in terms of the required computational time. However, the control problem in some cases does not change much from one sampling instance to the next, and therefore hot-start can be used to considerably accelerate the solution of an online optimisation problem. An adaptive optics system is used in this work as an example of such a control system to evaluate the benefits of hot-start for different families of optimisation algorithms. A comparison of the computational times and a discussion of hot-start efficiency for Interior Point, Active Set, Gradient-based and Augmented Lagrangian algorithms are provided in this contribution.

\section{I. Introduction}

The emergence of low-cost and fast computers has expanded the use of Model Predictive Control (MPC) into areas of engineering where the application of online constrained control was believed to be impossible. The main obstacle that limits the use of MPC is the computational complexity of solving an optimisation problem online. However, when the sampling rate is fast and the dynamics of the output disturbance is slower than the dynamics of the plant, one can substantially accelerate the solution of an online Quadratic Programming (QP) problem by using hot-start. By hot-start we mean the use of the solution and (partial) data from the previous time instant as a starting point for the current QP problem, instead of solving the optimisation problem from scratch (cold-start).

Despite the fact that the idea of hot-start is fairly straightforward, a careful selection of the optimisation algorithms is required. Evaluation of hot-start efficiency for different types of QP algorithms is complicated because the performance of QP solvers depends on the size of the problem and the percentage of active constraints. The information about hot-start efficiency is scattered across the literature \cite{1}–\cite{7}, and it is often difficult to relate the efficiency of hot-start between different QP algorithms.

This contribution provides numerical results of a comparison of hot-start efficiency for different families of QP algorithms. Using an adaptive optics (AO) closed-loop control scheme as a case-study, we compare and discuss the hot-start efficiency for Interior Point, Active Set (parametric and primal range-space), Gradient-based (projected alternating method), and Augmented Lagrangian (alternating directions method of multipliers) algorithms.

\section{II. Problem Formulation}

We use a standard control scheme of an AO system \cite{8} shown in Fig. 1 for evaluation of hot-start efficiency of different QP algorithms for MPC. We consider an AO system as a Linear Time-Invariant (LTI) model that is formulated to compensate an output disturbance, which is the atmospheric turbulence. We use the same formulation of MPC and the models of plant and disturbance as in \cite{8}, \cite{9}, and we briefly summarise them here for convenience.

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{Fig_1.png}
\caption{The overall control scheme of an AO system.}
\end{figure}

\subsection{A. The model of the deformable mirror}

The dynamics of the actuators is assumed to be a first-order transfer function (TF) with a fast pole. To account for the fact that in a hardware deformable mirror (DM) the dynamics of actuators varies from one actuator to another, we take the values of poles of continuous TFs from a Normal Distribution $p_{\mathcal{N}} \sim \mathcal{N}(p, \sigma_p^2)$ with mean $p$ and standard deviation $\sigma_p = \alpha \cdot p$. The TF for the $(i, \hat{i})$-th actuator is:

$$G(s)_{\text{plant}}^{i, \hat{i}} = \frac{1}{s + p_{\mathcal{N}}},$$

where the mean value of the poles is $p = 1500$ and $\alpha = 0.01$.

When an actuator is energised (pushed up or pulled down), it displaces neighbour actuators due to the sturdiness of the mirror’s surface. The fact that one actuator affects its neighbour actuators, which is called \textit{inter-actuator coupling}, makes the dynamics of a DM more complex. We consider only the case of coupling between four nearest neighbour actuators, which can be modelled as a proportional relationship between the actuators’ transfer functions and described

\footnotesize
\textsuperscript{1}Mikhail Konnik is with the School of Electrical Engineering and Computer Science, The University of Newcastle, Callaghan 2308, Australia. mail@mikonnik.info
\textsuperscript{2}José De Doná is with the Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan 2308, Australia. Jose.Dedona@newcastle.edu.au

The goal for a Quadratic Programming (QP) algorithm is to minimize a cost function with a constrained QP problem. The matrices $G_{\text{plan}}(s)$ and $G_{\text{plan}}(s)$ are defined as:

$$
G_{\text{plan}}(s) = \begin{bmatrix}
G_{1,1} & G_{1,2} & 0 & G_{1,4} & 0 & \ldots \\
G_{2,1} & G_{2,2} & G_{2,3} & 0 & G_{2,5} & \ldots \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\
\end{bmatrix}.
$$

That is, for an example of a $3 \times 3$ actuators grid (9 actuators), the coupling TF $G_{5,2}(s)$ between the actuators 5 and 2 is $G_{5,2}(s) = \gamma \cdot G_{5,5}(s)$, where $\gamma$ is the coupling degree. In this article we consider a coupling degree $\gamma = 0.2$.

### B. The model of the output disturbance

As one can see from the scheme in Fig. 1, atmospheric turbulence is the source of the output disturbance, which can be adequately described by an autoregressive model AR(2):

$$
G(s)_{i,i} = \frac{1}{s^2 + s \cdot \kappa_1 + \kappa_2}.
$$

The poles of the output disturbance are usually slower than the poles of a deformable mirror (DM). Here the mean values of the coefficients in (3) were set as $\kappa_1 = 2$ and $\kappa_2 = 30$.

### C. Complete model of the system dynamics

The continuous TFs of the plant $G_{\text{plan}}(s)$ in (1) and the TFs of atmospheric turbulence $G_{\text{atm}}(s)$ in (3) were assembled into transfer matrices such as (2) and converted to the state space formulation. The state space matrices in turn were discretised using a zero-order hold on the inputs with a sampling time of $T_s = 10^{-3}$ seconds and augmented to account for both plant and turbulence dynamics as follows:

$$
A = \begin{bmatrix}
A_{\text{plant}} & 0 \\
0 & A_{\text{atm}}
\end{bmatrix},
B = \begin{bmatrix}
B_{\text{plant}} \\
0
\end{bmatrix},
C = [C_{\text{plan}}C_{\text{atm}}].
$$

### D. Formulation of Model Predictive Control

We use the standard state space formulation of MPC [10] with a quadratic cost function:

$$
V_{N_p,N_c} = \frac{1}{2} U^T H U + U^T F \hat{x}_0,
$$

where $\hat{x}_0$ is the estimate of the current state obtained by a Kalman filter, $U = [u_0^T, u_1^T, \ldots, u_{N_p-1}^T]^T$ is a vector of future controls, $N_p$ is the state prediction horizon, $N_c$ is the control prediction horizon. The matrices $H$ and $F$ are defined as:

$$
H \triangleq \Gamma^T Q \Gamma + R, \quad F \triangleq \Gamma^T Q \Omega,
$$

where $Q = \text{diag}(Q^T Q, \ldots, P)$ and $R = \text{diag}(R, \ldots R)$. In turn, the matrices $Q = I$ and $R = 10^{-12} I$ are tuning parameters, and the matrices $\Gamma$ and $\Omega$ are defined as:

$$
\Gamma = \begin{bmatrix}
B & \ldots & 0 \\
AB & \ldots & 0 \\
\vdots & \ddots & \vdots \\
A^{N_p-1}B & \ldots & A^{N_p-N_c}B
\end{bmatrix},
\Omega = \begin{bmatrix}
A & A^2 & \ldots \\
\vdots & \ddots & \vdots \\
A^{N_p-1} & \ldots & A^{N_p-N_c}
\end{bmatrix}.
$$

The goal for a Quadratic Programming (QP) algorithm is to minimize the cost function (5):

$$
\min_U \frac{1}{2} U^T H U + U^T F \hat{x}_0 \quad \text{subject to } M \cdot U \leq \Lambda,
$$

where $M$ is the constraints matrix, and $\Lambda$ is the vector of maximum and minimum allowed values for the control input.

### E. The Problem

For certain types of control systems, such as adaptive optics systems, the use of hot-start leads to a significant speed-up in terms of the computation time. This is due to a fast sampling rate and slow dynamics of the output disturbance compared to the plant’s response. Therefore, the QP problem for MPC does not change much, in general, from one sampling instance to the next. In this case, hot-start is a key technique that allows to achieve extremely fast computation times for a constrained control.

The problem is that the benefits of hot-start vary for different types of QP algorithms and percentage of active constraints. This contribution provides a comparison of hot-start efficiency for the main families of QP algorithms.

### III. SIMULATION APPROACH

The hot-start efficiency for different QP algorithms was measured during the closed-loop operation of MPC. The constrained QP problem (8) was solved online, and two series of measurements for the computational time were made: 1) when the problem is solved from scratch (the cold-start case); and 2) when the previous solution is used as a starting point for solving the current problem (the hot-start case). We evaluated the efficiency of hot-start to show the dependency of computational time on the percentage of active constraints for different sizes of the control problem (number of actuators in the DM). The reported results are for the case of shortest prediction horizons $N_p = 2$ and $N_c = 1$.

### A. Simulation approach for triggering the active constraints

We use the model of the output disturbance from (3) driven by a white noise to generate $K = 10000$ instances of the disturbance, each instance is a vector of size $(n^2) \times 1$, where $n^2$ is the number of actuators of the DM (placed in an $n \times n$ actuators array). The sequence of the output disturbance was then re-used for all QP algorithms. The amplitude of the...
output disturbance was gradually increased to activate more and more constraints, as illustrated in Fig. 2 (middle subplot).

One can see in Fig. 2 (lower part of the plot) that the control signal has reached constraints at sampling instance 2100 and, consequently, the uncompensated output disturbance increases from this point (see Fig. 2, upper part of the plot). By increasing the amplitude of the output disturbance as illustrated in Fig. 2, we were able to gather enough data to evaluate the dependency of the computational time for a QP algorithm on the percentage of active constraints.

B. Representation of computational time for QP algorithms

Each series of numerical simulation generates a \( K \times 1 \) vector of computational time values \( T \) and a \( K \times 1 \) vector of the percentage of active constraints \( \Sigma \). We select all the elements of the vector \( T \) that correspond to a certain percentage of active constraints accumulated in the vector \( \Sigma \). That is, for 1% of active constraints we have \( k_1 \) values of computational times, \( k_2 \) values of time for 2% of active constraints and so on. The computational time has a probability distribution due to the randomness\(^3\) of the output disturbance, which results in more or less iterations required to solve a QP problem.

![Graph](image)

Fig. 3. Distribution of the computational time versus percentage of active constraints.

The distribution of time can be represented by a histogram (see Fig. 3, right part). To show the distribution of computational time for all percentages of active constraints, we made a scatter plot (see Fig. 3, left part), where each vertical group of dots shows the distribution of the computational time for the specific percentage of active constraints. The bigger the markers on the plot, the more probable that the calculations of the control signal will take a certain time. The computational time for QP algorithms was measured by the tic-toc MATLAB functions, which have a 1/\( \mu s \) resolution.

IV. COMPARISON OF HOT-START EFFICIENCY FOR QUADRATIC PROGRAMMING ALGORITHMS

The QP algorithms were compiled as MEX functions for MATLAB 2007b and run on a computer with Intel Core i7-2640M Processor 2.80GHz and 8 Gb DDR3 SDRAM under Debian GNU/Linux. We selected the QP algorithms from Interior Point, Active Set, Gradient-based, and Augmented Lagrangian families. The details on the implementation of the optimisation algorithms can be found in [9] (Chapter 5) and in the references provided below.

A. Interior Point algorithms

Interior point (IP) methods have emerged [11]–[13] as a strong competitor to active set methods, and are used in MPC [14]. Despite many attractive features, such as a guaranteed polynomial runtime and relatively constant computational time, there is no efficient\(^2\) warm start technique for IP methods (to the best of authors knowledge). This is because pre-processing of the hot-starting point is required for IP methods, which further complicates the resulting QP algorithm. That is, one cannot use the previous solution directly as a hot-start point for IP algorithms.

There were several attempts [2]–[7] to bring the hot-start capabilities into IP algorithms. For example, a shift strategy was proposed in [3], where the relative complementary slackness was used as the index for choosing the warm start point. A particular IP method, namely the Direct Non-linear Interior Point Algorithm (DNIPA) for optimal power flows used in [2] was shown to be efficient with hot-start, yielding an average iteration speed-up for hot-start of about 50%.

The algorithms based on Mehrotra Primal-Dual IP method [11], such as CVXGEN [13], do not use warm start because the algorithm’s performance does not improve significantly. A detailed discussion of IP hot-start is outside the scope of this article, but will be addressed in future work.

B. Primal Range-Space Active Set QP

The Range-Space Primal Active Set (RSPAS) [15] algorithm’s iterations are computed as \( U_{k+1} = U_k + \alpha_k p_k \), where \( \alpha_k \in (0,1] \) is a step-length, and the step direction \( p_k \) is found by solving the convex equality constrained subproblem

\[
\min_{p_k} \frac{1}{2} p_k^T H_k p_k + g_k^T p_k \quad \text{subject to} \quad M_k p_k = 0.
\]

Here \( g_k = (\bar{y} + \bar{y} U_k) \) is the gradient and the subset of constraints that are satisfied is represented by \( M_k \in \mathbb{R}^{m \times n} \). Hot-start for the RSPAS method uses the previous solution \( U_{k-1} \) as an initial starting point for \( U_k \), and re-uses matrices \( L_{k-1} \) and \( Y_{k-1} \) from the LQ factorisation \( MS^{-1} = LY^T \), \( H = S^T S \) at the previous sampling instance \( t_{k-1} \).

The advantages of hot-start for RSPAS algorithm are shown in Fig. 4. The computational time for the 7 \( \times \) 7 actuators DM model and the case of 20% active constraints was about 200 – 300\( \mu s \) for cold-start, while it takes only about 40\( \mu s \) in the hot-start mode to solve the same problem. Larger problems such as 10 \( \times \) 10 actuators DM can be solved within 100 – 150\( \mu s \) for the case of 20–40% of active constraints in the hot-start mode, while in cold-start it takes considerably longer than 1000\( \mu s \). Such a speed-up is due to the minor change in the active set because of fast sampling.

\(^2\)The difficulty with hot start in IP methods is that the optimal solution of the original problem satisfies the complementary condition for a convergence tolerance parameter, but unless it is optimal to the perturbed problem as well, it is either primal or dual infeasible, or both [2].

\(^3\)The spread of computational time can in part be attributed to background processes in the operation system taking up “wall clock” time. To minimise the impact of background processes, MATLAB was assigned to a designated core of the four-core processor.
thus requiring only one or two iterations for the RSPAS QP to find a new optimum.

C. Parametric Active Set qpOASES

The Parametric Active Set method, which is implemented in the qpOASES code [16], solves a parametric QP in the form: \( \min QP(w_0) = \frac{1}{2} U^T H U + U^T g(w_0) \) subject to \( Gx \geq b(w_0) \), where \( G \) is a constraint matrix. To transition from a solved quadratic program \( QP(w_0) \) to the next one \( QP(w_0^{\text{new}}) \), the algorithm moves on a straight line in the parameter space [17]. The details of the hot-start implementation of the qpOASES_sequenceSB algorithm can be found in [16], [17]. If no initial guess of the starting point is provided, iterations start from the origin [1].

Similar to the RSPAS algorithm in Subsection IV-B, the hot-start operation mode is highly beneficial for the qpOASES code. In Fig. 5 we present a comparison of a cold-started version qpOASES and the hot-started version qpOASES_sequenceSB. The hot-start for the qpOASES algorithm speeds up the computations significantly: up to 30x times for the 7 \( \times \) 7 actuators DM and up to 80x times for the larger problem of 10 \( \times \) 10 actuators DM, according to Fig. 5. That is, in the hot-start operation of qpOASES, one can solve a 7 \( \times \) 7 actuators DM problem within about 55–60\( \mu \)s (stays almost the same for any percentage of the active constraints), and within 140–180\( \mu \)s for a larger control problem of 10 \( \times \) 10 actuators DM.

A remarkable property of the Parametric Active Set homotopy-based [18] methods is that the computational time does not increase with the number of active constraints, which is similar to IP methods. This is a highly desirable property of a QP algorithm since it makes the behaviour of the solver predictable in terms of the computational time.

D. Projected Alternating Barzilai-Borwein (PABB)

The Barzilai and Borwein (BB) gradient-based method [19] combines ideas from the steepest descent and Quasi-Newton methods. Iterations of the Projected Alternating Barzilai-Borwein (PABB) algorithm are computed as \( U_{k+1} = P_U [U_k - \alpha_k^{BB} g_k] \), where \( g_k = (P + \mathbb{H} U_k) \) is the gradient vector and \( P_U \) the projection operator onto the feasible set \( \Omega = \{ U \in \mathbb{R}^m : U_{\text{min}} \leq U \leq U_{\text{max}} \} \) with \( U_{\text{min}} \) and \( U_{\text{max}} \) denoting the lower and upper bounds of the control inputs. The step length \( \alpha_k^{BB1} = \frac{s_k \cdot s_k - s_{k-1} \cdot s_{k-1}}{s_k \cdot y_k + s_{k-1} \cdot y_{k-1}} \) or \( \alpha_k^{BB2} = \frac{s_k \cdot y_k - s_{k-1} \cdot y_{k-1}}{s_k \cdot y_k + s_{k-1} \cdot y_{k-1}} \) is computed via \( s_k = U_k - U_{k-1} \) and \( y_k = g_k - g_{k-1} \). The algorithm terminates when \( \tau \leq \| x_k \|_2 - \| x_{k-1} \|_2 \).
Hot-start in the PABB algorithm uses the gradient vector $g_k$ and the optimal value $U_{k-1}$ from the previous sampling instance $t_{k-1}$ as a starting point for the current solution. The details can be found in [9] in Chapter 5.

The performance of the PABB algorithm (and the gradient-based algorithms in general) is affected by the tolerance value $\tau$ and the condition number of the Hessian matrix in (6). In this work the tolerance was set to $\tau = 10^{-4}$ and the condition number of the Hessian matrix $\kappa(H) \sim 10^2 \ldots 10^4$.

The hot- and cold-start modes are compared in Fig. 6 for $7 \times 7$ and $10 \times 10$ actuators DMs. Expectedly, the hot-started PABB is considerably faster than cold-started: we observe $1.5 - 2x$ less iterations, and about $2x$ faster computation time. The acceleration is more pronounced for the unconstrained case, where it can be up to $2 - 3x$ depending on the condition number and the value of tolerance $\tau$.

The convergence of the PABB algorithm deteriorates as the condition number of the Hessian matrix increases, thus the spread of computational time grows. Such a decrease in the performance with the growth of the condition number is typical for the gradient-based QP algorithms.

E. Alternating Directions Method of Multipliers (ADMM)

The ADMM QP algorithm is a form of Augmented Lagrangian method, which is suitable for large-scale problems and distributed computational environments [20]. An optimisation problem for the ADMM algorithm can be rewritten [21] in the dual form: $\min f(x) + g(z)$ subject to $x - z = 0$ with variable $x \in \mathbb{R}^n$, where $f(x) = \frac{1}{2}x^T H x + q^T x$ and the constraints set $\Omega$ are convex, and $g$ is the indicator function of $\Omega$. Iterations of the ADMM algorithm consist of: (1) an $x$-minimisation step $x^{k+1} = \arg \min (f(x) + (\rho/2)\|x-z^k+u^k\|_2^2)$; (2) a $z$-minimisation step $z^{k+1} = P_\Omega (x^{k+1} + u^k)$, where $P_\Omega$ is the projection operator onto the feasible set $\Omega$; and (3) a dual variable update $u^{k+1} = u^k + z^{k+1} - z^k$.

The primal and dual residuals are $r^k = x^k - z^k$ and $s^k = -\rho(z^k - z^{k-1})$, where $\rho$ is the Augmented Lagrangian parameter. The ADMM algorithm stops when $\|r^k\|_2 \leq \tau$ or $\|s^k\|_2 \leq \tau$, where $\tau$ is a prescribed tolerance.

Hot-start in ADMM is implemented by feeding the vectors $x^k$, $u^k$ and $z^k$ from the previous iteration $t_{k-1}$ to the current iteration. The details of ADMM can be found in [20]–[24].

The results for the ADMM algorithm are presented Fig. 7.
for 1000 Hz sampling rate and the tolerance $\tau = 10^{-4}$. The hot-start operation mode in ADMM reduces the number of iterations by 1.4 – 1.7 times compared to cold-start, as seen in Fig. 7(a). For the $7 \times 7$ actuators DM case, the acceleration due to hot-start is about 1.8x times for the unconstrained case and about 1.5-1.7x times for the constrained case.

The benefits of hot-start for the ADMM depend on the number of active constraints, with a tendency to decrease as the number of active constraints grows (i.e., hot-start is more efficient when there are few active constraints). A reasonable efficiency of hot-start along with the simplicity of the algorithm (about 70 lines of C code) makes ADMM QP an attractive choice for fast MPC.

V. RESULTS DISCUSSION AND CONCLUSIONS

In this contribution we have evaluated and discussed the efficiency of hot-start operation for different families of QP algorithms that can be used for fast MPC. Hot-start in QP is a key technique for achieving extremely fast computation times in MPC when the sampling rate is fast and the disturbance does not change much from one sampling instance to the next. The results of simulations, which are based on an adaptive optics case study, can be summarised as:

- **Interior Point methods**: hot-start is possible but not efficient, therefore for many IP algorithms (such as e.g. CVXGEN [13]) there is no noticeable speed-up. This is because pre-processing of the hot-starting point is required for IP methods that further complicates the resulting QP algorithm. However, certain IP algorithms [2] were reported to have the iteration speed-up for hot-start of about 50%.

- **Active Set methods**: hot-start is not hard to implement, and the benefits are substantial. Depending on the algorithm, the speed-up can be from 10-30x up to 80x. Certain algorithms, such as qPOASES, are based on the use of a homotopy for parametric QP [18] and have an almost constant computational time with the percentage of active constraints - a highly desirable property for embedded control.

- **Gradient-based methods**: hot-start is efficient and easy to implement, although the speed-up depends on the tolerance and the conditioning number of the Hessian matrix. One can expect 1.5 – 2x less iterations and about 2x faster computation time in the hot-start mode. The acceleration is higher for the unconstrained case.

- **Augmented Lagrangian methods**: hot-start is easy to implement, but the benefits are marginal with a tendency to decrease as the number of active constraints grows. The number of hot-start iterations were reduced by 1.4 – 1.7 times compared to cold-start for the ADMM QP.

To sum up, we conclude that the Active Set QP algorithms benefit the most from hot-start: it yields a substantial speed-up for the constrained case and is easy to implement. The next step will be to investigate which QP methods might benefit from a parallel implementation (e.g., on GPU).

REFERENCES


