Reducing the Prediction Horizon in NMPC: An Algorithm Based Approach

J. Pannek * K. Worthmann **

* Curtin University of Technology, Perth, 6845 WA, Australia
  (e-mail: juergen.pannek@googlemail.com)
 ** University of Bayreuth, 95440 Bayreuth, Germany
  (e-mail: karl.worthmann@uni-bayreuth.de)

Abstract: In order to guarantee stability, known results for MPC without additional terminal costs or endpoint constraints often require rather large prediction horizons. Still, stable behavior of closed loop solutions can often be observed even for shorter horizons. Here, we make use of the recent observation that stability can be guaranteed for smaller prediction horizons via Lyapunov arguments if more than only the first control is implemented. Since such a procedure may be harmful in terms of robustness, we derive conditions which allow to increase the rate at which state measurements are used for feedback while maintaining stability and desired performance specifications. Our main contribution consists in developing two algorithms based on the deduced conditions and a corresponding stability theorem which ensures asymptotic stability for the MPC closed loop for significantly shorter prediction horizons.

Keywords: model predictive control, stability, suboptimality estimate, algorithms, sampled data system

1. INTRODUCTION

Model predictive control (MPC), sometimes also termed receding horizon control (RHC), deals with the problem of approximately solving an infinite horizon optimal control problem which is, in general, computationally intractable. To this end, a solution of the optimal control problem on a truncated, and thus finite, horizon is computed. Then, the first part of the resulting control is implemented at the plant and the finite horizon is shifted forward in time which renders this method to be iteratively applicable, see, e.g., Magni et al. (2009).

During the last decade, theory of MPC has grown rather mature for both linear and nonlinear systems as shown in Allgöwer and Zheng (2000) and Rawlings and Mayne (2009). Additionally, it is used in a variety of industrial applications, cf. Badgwell and Qin (2003) due to its capability to directly incorporate input and state constraints. However, since the original problem is replaced by an iteratively solved sequence of control problems – which are posed on a finite horizon – stability of the resulting closed loop may be lost. In order to ensure stability, several modifications of the finite horizon control problems such as stabilizing terminal constraints or a local Lyapunov function as an additional terminal weight have been proposed, cf. Keerthi and Gilbert (1988) and Chen and Allgöwer (1999).

Here, we consider MPC without additional terminal constraints or costs, which is especially attractive from a computational point of view. Stability for these schemes can be shown via a relaxed Lyapunov inequality, see Grüne (2009) for a set valued framework and Grün and Pannek (2009) for a trajectory based approach. Still, stable behavior of the MPC controlled closed loop can be observed in many examples even if these theoretical results require a longer prediction horizon in order to guarantee stability. The aim of this paper consists of closing this gap between theory and experimental experience. In particular, we develop an algorithm which enables us to ensure stability for a significantly shorter prediction horizon and, as a consequence, reduces the computational effort in each MPC iteration significantly. The key observation in order to extend the line of arguments in the stability proofs proposed in Grün (2009) is that the relaxed Lyapunov inequality holds true for the MPC closed loop for smaller horizons if not only the first element but several controls of the open loop are implemented, cf. Grün et al. (2010). However, this may be harmful in terms of robustness, cf. Magni and Scattolini (2007). Utilizing the internal structure of consecutive MPC problems in the critical cases along the closed loop trajectory, we present conditions which allow us to close the loop more often and maintain stability. Indeed, our results are also applicable in order to deduce performance estimates of the MPC closed loop.

The paper is organized as follows: In Section 2, the MPC problem under consideration is stated. In the ensuing section, we extend suboptimality results from Grüne and Pannek (2009) and Grüne et al. (2010) in order to give a basic MPC algorithm which checks these performance bounds. In Section 4, we develop the proposed algorithm further in order to increase the rate at which state measurements are used for feedback and state our main stability theorem. In the final Section 5 we draw some conclusions. Instead
MPC consists of three steps which are repeated at every control (MPC) approach instead. The fundamental idea of using Bellman’s principle of optimality. However, since the state space X and the control value space U are arbitrary metric spaces. As a consequence, the following results are applicable to discrete time dynamics induced by a sampled – finite or infinite dimensional – system, see, e.g., Altmüller et al. (2010) or Ito and Kunisch (2002). Constraints may be implemented at the plant.

Example 2.2. Consider the synchronous generator model
\[
\begin{align*}
\dot{x}_1(t) &= x_2(t) \\
\dot{x}_2(t) &= -b_1 x_2(t) \sin x_1(t) - b_2 x_1(t) + P \\
\dot{x}_3(t) &= b_3 \cos x_1(t) - b_4 x_3(t) + E + u(t)
\end{align*}
\]
with parameters \(b_1 = 34.29, b_2 = 0.0, b_3 = 0.149, b_4 = 0.3341, P = 28.22, \) and \(E = 0.2405\) from Galaz et al. (2003) and the initial value \(x_0 = (1.02, 0.1, 1.014)\).

The key assumption in Proposition 2.1 is the relaxed Lyapunov–inequality (6) in which \(\alpha\) can be interpreted as a lower bound for the rate of convergence. From the literature, it is well-known that this condition is satisfied for sufficiently long horizons \(N\), cf. Jadabae and Hauser (2005), Grimm et al. (2005) or Alamir and Bornard (1995), and that a suitable \(N\) may be computed via methods described in Pannck (2009). To investigate whether this condition is restrictive, we consider the following example.

Example 2.2. Consider the synchronous generator model
For Example 2.2 our aim is to steer the system to its equilibrium $x^* \approx (1.12, 0.0, 0.914)$. To this end, we use MPC to generate results for sampling time $T = 0.1$ and running cost

$$
\ell(x, u) = \int_0^T \| \varphi(t; x, \tilde{u}) - x^* \|^2 + \lambda \| \tilde{u}(t) \|^2 dt
$$

with piecewise constant control $\tilde{u}(t) = u$ for $t \in [0, T]$ and $\lambda = 10^{-6}$ where $\varphi(\cdot; x, \tilde{u})$ denotes the solution operator of the differential equation with initial value $x$ and control $\tilde{u}$. For this setting, we obtain stability of the closed loop via inequality (6) for $N \geq 30$, cf. Figure 1. However, Figure 1 shows stable behavior of the closed loop for the smaller prediction horizon $N = 19$ without significant deviations – despite the fact that the relaxed Lyapunov–inequality from Proposition 2.1 is violated at three instances. Our numerical experiment shows that, even if stability and the desired performance estimate (7) cannot be guaranteed via Proposition 2.1, stable and satisfactory behavior of the closed loop may still be observed. Hence, since the computational effort grows rapidly with respect to the prediction horizon, our goal consists of developing an algorithm which allows for significantly reducing the prediction horizon. To this end, we adapt the theoretical condition given in (6).

In Grüne et al. (2010), it has been shown that the suboptimality estimate $\alpha$ is increasing (up to a certain point) if more than one element of the computed sequence of control values is applied. Hence, we vary the number of open loop control values to be implemented during runtime at each MPC iteration, i.e., the system may stay in open loop for more than one sampling period, in order to deal with this issue. Doing so, however, may lead to severe problems, e.g., in terms of robustness, cf. Magni and Scattolini (2007). Hence, the central aim of this work is the development of an algorithm which checks – at runtime – whether it is necessary to remain in open loop in order to guarantee the desired stability behavior of the closed loop by suboptimality arguments, or if the loop can be closed without loosing these characteristics.

To this end, we introduce the list $S = (s(0), s(1), \ldots) \subseteq \mathbb{N}_0$, which we assume to be in ascending order, in order to indicate time instances at which the control sequence is updated. Moreover, we denote the closed loop solution at time instant $s(n)$ by $x_n = x_{\mu_n} (s(n))$ and define $m_n := s(n + 1) - s(n)$, i.e., the time between two MPC updates. Hence,

$$
x_{\mu_n} (s(n) + m_n) = x_{\mu_{n+1}} (s(n + 1)) = x_{n+1}
$$

holds. This enables us – in view of Bellman’s principle of optimality – to define the closed loop control

$$
\mu_{N, m_n}^S(\cdot; x_n) := \arg\min_{u \in \mathbb{U}^{m_n}} \left\{ V_{N-m_n}^S(x_n(m_n; x_n)) \right\}
$$

and the infinite horizon optimal value function $V^*_\infty(\cdot)$. In particular, it has been shown that given a controllability condition, i.e., for each $x_0$ there exists a control $u_{x_0}$ such that

$$
\ell(x_{u_{x_0}(n); x_0}, u_{x_0}(n; x_0)) \leq C\sigma^n \min_{\forall u \in \mathbb{U}} \ell(x_0, u)
$$

holds with $C \geq 1$, $\sigma \in (0, 1)$, then there exists a prediction horizon $N \in \mathbb{N}_{\geq 2}$ and $\overline{m} \in \{1, \ldots, [N/2]\}$ such that $\alpha_{N, \overline{m}} \geq \overline{\pi}$ for an arbitrarily specified $\overline{\pi} \in (0, 1)$.

Example 3.1. Suppose (10) to hold with $C = 4$ and $\sigma = 0.6$ and fix $\overline{\pi} = 0.275$. Then the pair $(N, \overline{m}) = (15, 6)$ implies $\alpha_{N, \overline{m}} \approx 0.294 > \overline{\pi}$, cf. Figure 2, i.e., the closed–loop satisfies the performance bound $\overline{\pi}$. However, $N = 25$ is the smallest prediction horizon which yields $\alpha_{N, m} \geq \overline{\pi}$ for $m = 1$, i.e., classical MPC, cf. Figure 3.

Examples 3.1 shows that choosing a larger value for $m$ may significantly reduce the required prediction horizon length and, consequently, the numerical effort which grows rapidly with respect to the horizon $N$.

Since the shape of the curve in Figure 2 is not a coincidence, i.e., $\alpha_{N, [N/2]} \geq \alpha_{N,m}$ holds for $m \in \{1, \ldots, N-1\}$, cf. Grüne et al. (2010), we may easily determine the small-
Reordering (11), we obtain
\[ n \sum_{i=0}^{m_n-1} \ell(x_{u_N}(k; x_i), u_N(k; x_i)) \leq V_N(x_0) - V_N(x_{n+1}) \leq V_N(x_0) \]
and hence taking \( n \) to infinity implies the assertion. \( \square \)

**Remark 3.3.** Note that the assumptions of Proposition 3.2 indeed imply the estimate
\[ V_\infty(x_n) \leq V_\infty^m(x_n) \leq \frac{1}{\alpha} V_\infty(x_n) \] (13)
for all \( n \in \mathbb{N}_0 \) which can be proven analogously.

Note that \( S \) is built up during runtime of the algorithm and not known in advance. Hence, \( S \) is always ordered. A corresponding implementation which aims at guaranteeing a fixed lower bound of the degree of suboptimality \( \bar{\alpha} \) takes the following form:

Given state \( x := x_0 \), list \( S = (0) \), \( N \in \mathbb{N}_{\geq 2} \), and \( \bar{\alpha} \in (0, 1) \)

1. Set \( j = 0 \), compute \( u_N(; x) \) and \( V_N(x) \). Do
   a. Set \( j = j + 1 \), compute \( V_N(x_{u_N}(j; x)) \)
   b. Compute maximal \( \alpha \) to satisfy (11)
   c. If \( \alpha \geq \bar{\alpha} \), set \( m_n = j \) and goto 2
   d. If \( j = N - 1 \): Print warning “Solution may diverge”, set \( m_n = 1 \) and goto 2 while \( \alpha < \bar{\alpha} \)
2. For \( j = 1, \ldots, m_n \) do
   Implement \( \mu_{\bar{\alpha}}^j (j-1; x) := u_N(j-1; x) \)
3. Set \( S := (S, \text{back}(S) + m_n), x := x_{\mu_{\bar{\alpha}}^j}(m_n; x) \), goto 1

**Remark 3.4.** Here, we adopted the programming notation back which allows for fast access to the last element of a list.

**Remark 3.5.** If (11) is not satisfied for \( j \leq N - 1 \), a warning will be printed out because the performance bound \( \bar{\alpha} \) cannot be guaranteed. In order to cope with this issue, there exist remedies, e.g., one may increase the prediction horizon and repeat step 1. For sufficiently large \( N \), this ensures the local validity of (11). Unfortunately, the proof of Proposition 3.2 cannot be applied in this context due to the prolongation of the horizon. Yet, it can be replaced by estimates from Pannek (2009) or Giselsson (2010) for varying prediction horizons to obtain a result similar to (12). Alternatively, one may continue with the algorithm. If there does not occur another warning, the algorithm guarantees the desired performance for \( x_n \) instead of \( x_0 \), i.e., from that point on, cf. Remark 3.3.

In order to obtain a robust procedure, however, it is preferable that the control loop is closed as often as possible, i.e., \( m_n = 1 \) for all \( n \in \mathbb{N}_0 \). In the following section, we give results and corresponding algorithms which – given certain conditions – allow us to reduce \( m_n \).

4. ALGORITHMS AND STABILITY RESULTS

In many applications a stable behavior of the closed loop is observed for \( m_n = 1 \) even if it cannot be guaranteed using the suboptimality estimate (6), cf. Example 2.2. Here, we present a methodology to close the control loop more often while maintaining stability. In particular, if \( m_n > 1 \) is required in order to ensure \( \alpha \geq \bar{\alpha} \) in (11) and, as a
consequence, in step (1) of the proposed algorithm, we present conditions which allow us to update the control for some \( j \in \{1, \ldots, m_n - 1\} \) to

\[
\hat{u}_{N}(k; x_n) := \begin{cases} 
    u_N(k; x_n), & k \leq j - 1 \\
    u_N(k - j; x_n(j; x_n)), & k \geq j
\end{cases}
\]  

(14)

insert the new updating time instant \( s(n) + j \) into the list \( S \), and still be able to guarantee the desired stability behavior of the closed loop. Summarizing, our aim consists of guaranteeing that the degree of suboptimality \( \pi \) is maintained, i.e., (11) is still satisfied, for the MPC law updated at a time instant \( j \leq m_n - 1 \) according to (14).

**Proposition 4.1.** Let \( u_{N}(:, x_n) \) and inequality (11) hold for \( u_{N}(:, x_n) \), \( \pi > 0 \), and \( m_n \in \{2, \ldots, N - 1\} \). If the inequality

\[
V_N(x_{u_N}(m_n - j; x_n(j; x_n))) - V_{N-j}(x_{u_N}(j; x_n))
\]

\[
\leq (1 - \alpha) \sum_{k=0}^{j-1} \ell(x_{u_N}(k; x_n), u_N(k; x_n)) - \alpha \sum_{k=j}^{m_n-1} \ell(x_{u_N}(k - j; x_n(j; x_n)), u_N(k - j; x_n(j; x_n))
\]

(15)

holds for some \( j \in \{1, \ldots, m_n - 1\} \), then the control sequence \( u_{N}(:, x_n) \) can be replaced by (14) and suboptimality degree \( \pi \) is locally maintained.

**Proof:** In order to show the assertion, we need to show (11) for the modified control sequence (14). Reformulating (15) by shifting the running costs associated with the unchanged control to the left hand side of (15) we obtain

\[
V_N(x_{u_N}(m_n - j; x_n(j; x_n))) - V_N(x_{u_N}(0; x_n))
\]

\[
\leq -\alpha \sum_{k=0}^{m_n-1} \ell(x_{u_N}(k; x_n), \hat{u}_{N}(k; x_n))
\]

which is equivalent to

\[
V_N(x_n) \geq V_N(x_{u_N}(m_n; x_n))
\]

\[
+ \pi \sum_{k=0}^{m_n-1} \ell(x_{u_N}(k; x_n), \hat{u}_{N}(k; x_n)),
\]

(16)

i.e., the relaxed Lyapunov inequality (11) for the updated control \( \hat{u}_{N}(:, x_n) \). \( \square \)

**Example 4.2.** Consider the illustrative Example 2.2 with horizon \( N = 19 \) and \( \pi = 0.1 \). For this setting, we obtain \( \alpha \geq \pi \) with \( m_n = 1 \) for all updating instances \( s_n \in S \) except for those three points indicated by \( [\square] \) in Figure 1 where we have \( m_n = 2 \) once and \( m_n = 3 \) twice. Yet, for each of these points, inequality (15) holds with \( j = 1 \).

- \( m_n = 2 \): We apply Theorem 4.1 in order to update the control \( u_{N}(:, x_n) := \hat{u}_{N}(:, x_n) \) according to (14). Thus, since the MPC law was updated at each sampling instant, we have obtained stability of the classical MPC closed loop.
- \( m_n = 3 \): We utilize Theorem 4.1 in an iterative manner: since (15) holds for \( j = 1 \), we proceed as in the case \( m_n = 2 \), update the control \( u_{N}(:, x_n) := \hat{u}_{N}(:, x_n) \), and compute \( x_{u_N}(2; x_n) \). Then, we compute \( u_{N}( :, x_{u_N}(2; x_n)) \) and check whether or not (15) holds for (14) with \( j = 2 \). Note that here, \( u_{N}( :, x_n) \) in (14) is already an updated control sequence. Since (15) holds true, we obtain stability of the closed loop for the updated MPC control sequence. Moreover, we applied \( \mu_{N}(x(s(n)), \mu_{N}(x(s(n)+1)) \) and \( \mu_{N}(x(s(n)+2)) \) instead of \( \mu_{N}^{2}(i; x(s(n))) \), \( i = 0, 1, 2 \), i.e., we have again used classical MPC.

Note that this iterative procedure can be extended in a similar manner for all \( m_n \leq N - 1 \).

Integrating the results from Theorem 4.1 into the algorithm displayed after Proposition 3.2 can be done by changing Step (2).

(2) For \( j = 1, \ldots, m_n \) do

- (a) Implement \( \mu_{N}^{\hat{S}}(j - 1; x) := u_{N}(j - 1; x) \)
- (b) Compute \( u_{N}( :, x_{u_N}(j; x)) \) and \( V_N(x_{u_N}(m_n; x)) \)
- (c) If \( j < m_n \) and condition (15) holds:
  - Construct \( \hat{u}_{N}( :, x) \) according to (14)
  - Update \( u_{N}( :, x) := \hat{u}_{N}( :, x) \).

Note that \( V_{N-j}(x_{u_N}(j; x_n)) \) in (15) is known in advance from \( V_N(x_n) \) due to the principle of optimality. Hence, only \( u_{N}( :, x_{u_N}(j; x)) \) and \( V_N(x_{u_N}(m_n; x)) \) need to be computed. Unfortunately, this result has to be checked for all \( j \in \{1, \ldots, m_n - 2\} \) using the comparison basis \( V_N(x_n) \). As a consequence, we always have to keep the updating instant \( s(n) \) in mind, cf. Example 4.2. The following result allows us to weaken this restriction:

**Proposition 4.3.** Let \( x_n \in \mathcal{X} \) be given and inequality (11) hold for \( u_{N}( :, x_n) \), \( \pi > 0 \), and \( m_n \in \{2, \ldots, N - 1\} \). If the inequality

\[
V_N(x_{u_N}(m_n - j; x_n(j; x_n))) - V_N(x_{u_N}(m_n; x_n))
\]

\[
\leq -\pi \sum_{k=0}^{m_n-1} \ell(x_{u_N}(k; x_n), u_N(k; x_n))
\]

\[
- \ell(x_{u_N}(k - j; x_n(j; x_n)), u_N(k - j; x_n(j; x_n))
\]

(17)

holds for some \( j \in \{1, \ldots, m_n - 1\} \), then the control sequence \( u_{N}( :, x_n) \) can be replaced by (14) and suboptimality degree \( \pi \) is locally maintained.

**Proof:** Adding (17) to (11) leads to

\[
V_N(x_{u_N}(m_n; x_n)) - V_N(x_n)
\]

\[
\leq -\pi \sum_{k=0}^{m_n-1} \ell(x_{u_N}(k; x_n), u_N(k; x_n))
\]

\[
- \pi \sum_{k=j}^{m_n-1} \ell(x_{u_N}(k; x_n), \hat{u}_{N}(k; x_n)).
\]

Updating the control sequence according to (14), we obtain (16) which completes the proof. \( \square \)

**Remark 4.4.** We also like to point out that conditions (15) and (17) allow for a less fast decrease of energy along the closed loop, i.e., the case \( V_N(x_{u_N}(m_n; x_n)) \geq V_N(x_{u_N}(m_n; x_n)) \) is not excluded in general.

As outlined in Example 4.2, Theorem 4.1 (and analogously Theorem 4.3) may be applied iteratively. Yet, in contrast to (15), \( V_N(x_n) \) is not required in (17). Hence, if an appropriate update can be found for time instant \( j \in \{1, \ldots, m_n - 1\} \), the loop can be closed and, as a consequence, Theorem 4.3 can be applied to the new initial value \( x_{u_N}(j; x_n) \) with respect to the reduced number...
of control to be implemented \( m_n - j \). To this end, the following modification of Step (2c) can be used:

\[(2c) \text{ If } j < m_n \text{ and condition (17) holds:}
\]
Construct \( \hat{u}_N(\cdot;x) \) according to (14)
Update \( u_N(\cdot;x) := \hat{u}_N(\cdot;x) \)
Update the values \( V_N(x_{u_n}(m_n;x)) \) and \( \ell(x_{u_n}(k;x), u_N(k;x)) \), \( k = j, \ldots, m_n - 1 \)

Combining the results from Proposition 3.2 and Theorems 4.1, 4.3, we obtain our main stability result:

**Theorem 4.5.** Let \( x_0 \in \mathcal{X}, \pi > 0 \) be given and apply one of the three proposed algorithms. Assume that (1c) is satisfied for some \( j \in \{1, \ldots, N - 1\} \) for each iterate. Then the closed loop trajectory satisfies the performance estimate (12) from Proposition 3.2. If, in addition, the conditions of Proposition 2.1(ii) hold, then \( x_{u,\pi}(\cdot) \) behaves like a trajectory of an asymptotically stable system.

**Proof:** The algorithms construct the set \( \mathcal{S} \). Since (1c) is satisfied for some \( j \in \{1, \ldots, N - 1\} \) the assumptions of Proposition 3.2, i.e., inequality (11) for \( x_n = x_{u,\pi}(s(n)) \), are satisfied which implies (12). To conclude asymptotically stable behavior of the closed loop, standard direct Lyapunov techniques can be applied. \( \square \)

**Example 4.6.** Again consider Example 2.2 with \( N = 19 \) and \( \pi = 0.1 \). Since (17) is more conservative than (15), the closed loop control and accordingly the closed loop solution \( x_{u,\pi}(\cdot) \) will not coincide in general. This also holds true in our example: While the case \( m_n = 2 \) can be resolved similarly using the algorithm based on (17), no update of the control \( u_N \) is performed for the two cases \( m_n = 3 \). Here, this does not cause any change in the list \( \mathcal{S} \). However, in general, \( \mathcal{S} \) is likely to be different for the presented algorithms.

Since the proposed algorithms guarantee stability for the significantly smaller horizon \( N = 19 \), the computing time for Example 2.2 is reduced by almost 50% – despite the additional computational effort for Step (2).

5. CONCLUSION

Based on a detailed analysis of a relaxed Lyapunov inequality, we have deduced conditions in order to close the gap between theoretical stability results and stable behavior of the MPC closed loop. Using this additional insight, we developed two MPC algorithms which check these conditions at runtime. Although the additional computational effort is not negligible, the proposed algorithms may allow us to shorten the prediction horizon used in the optimization. This reduces, in general, the complexity of the problem significantly which may imply a reduction of the overall computational costs.

**REFERENCES**


