Simple and efficient moving horizon estimation based on the fast gradient method

Bruno Morabito * **Markus Kögel** ** **Eric Bullinger** **
Gabriele Pannocchia * **Rolf Findeisen** **

* *Department of Civil and Industrial Engineering - Chemical Engineering Section. University of Pisa, Italy. e-mail: br.morabito@gmail.com, gabriele.pannocchia@unipi.it.*
** *Institute for Automation Engineering, Otto-von-Guericke-University Magdeburg, Germany. e-mail: {markus.koegel, eric.bullinger, rolf.findeisen}@ovgu.de.*

Abstract: By now many results with respect to the fast and efficient implementation of model predictive control exist. However, for moving horizon estimation, only a few results are available. We present a simple solution algorithm tailored to moving horizon estimation of linear, discrete time systems. In a first step the problem is reformulated such that only the states remain as optimization variables, i.e. process and measurement noise are eliminated from the optimization problem. This reformulation enables the use of the fast gradient method, which has recently received a lot of attention for the solution of model predictive control problems. In contrast to the model predictive control case, the Hessian matrix is time-varying in moving horizon estimation, due to the time-varying nature of the arrival cost. Therefore, we outline a tailored method to compute online the lower and upper eigenvalues of the Hessian matrix required by the here considered fast gradient method. In addition, we discuss stopping criteria and various implementation details. An example illustrates the efficiency of the proposed algorithm.

Keywords: Moving horizon estimation, State estimation, Fast gradient methods.

1. INTRODUCTION

State estimation plays a fundamental role in many applications. It is often elementary for monitoring a system and frequently utilized in combination with a state feedback controller to stabilize a system. Estimation methods based on Kalman filtering, compare Kailath et al. (2000), which are predominantly used in applications, do not allow to easily include constraints on the variables in a structured way. Therefore, Moving Horizon Estimation (MHE) has received increasing interest since it can effectively take constraints on the variables into account. MHE is, similarly as its control "relative" Model Predictive Control (MPC), based on the online solution of an optimization problem over a finite horizon. Therefore, the challenge of an efficient real-time implementation arises, especially for high sampling rates, limited computation power or large scale systems.

In comparison to fast MPC there are only limited results with respect to the efficient implementation of MHE available.

For fast MHE, different approaches have been investigated based on tailored solution approaches for the underlying optimization problem. For nonlinear systems results based on combinations of direct multiple shooting with Gauss-Newton iterations, see (Diehl et al., 2005; Kraus et al., 2006), or sensitivity analysis and nonlinear programming, see e.g. Zavala et al. (2008), exist. Also approximation based methods utilizing for example singular value decompositions, see Jang et al. (2014), or in situ adaptive tabulation, see Abrol and Edgar (2011), have been considered.

Darby and Nikolaou (2007) implemented a look-up table and function evaluation for real-time implementation of MHE for linear systems. Similarly to MPC, however, the number of polytopes generated in the approach tends to grow combinatorially with the number of constraints, which limits the size of the problem that can be handled. Haverbeke et al. (2009) developed a primal barrier interior-point method algorithm for linear system exploiting the system structure.

For moving horizon estimation of constrained, linear, time-discrete systems we propose a simple, tailored algorithm based on Nesterovs fast gradient method (Nesterov, 1983, 2004). Fast gradient methods have recently received considerable attention for the solution of optimization problems arising in model predictive control, see e.g. Faulwasser et al. (2014); Jerez et al. (2014); Kögel and Findeisen (2011); Patrinos and Bemporad (2014); Richter et al. (2012, 2010); Zometa et al. (2012). To enable the efficient solution of the MHE problem using the fast gradient method we propose to eliminate the optimization variables related to the noise from the optimization problem. This results in a sparse formulation with only the states as optimization variables. In contrast to MPC, in MHE the Hessian matrix and thus also its eigenvalues are time-varying due to the arrival cost. Since the fast gradient method requires the largest and smallest eigenvalues of the Hessian matrix (or tight bounds on them), we discuss how to efficiently compute these based on the so-called inverse iteration, (Golub and Van Loan, 2012). Additionally, we review stopping criteria and discuss the implementation of the proposed algorithm. The applicability and performance of the presented method is illustrated by an example. To enable the efficient solution of the MHE problem using the

fast gradient method we propose to eliminate the optimization variables related to the noise from the optimization problem. This results in a sparse formulation with only the states as optimization variables. In contrast to MPC, in MHE the Hessian matrix and thus also its eigenvalues are time-varying due to the arrival cost. Since the fast gradient method requires the largest and smallest eigenvalues of the Hessian matrix (or tight bounds on them), we discuss how to efficiently compute those based on the so-called inverse iteration, compare Golub and Van Loan (2012). Additionally, we review stopping criteria and discuss the implementation of the proposed algorithm. The applicability and the performance of the presented method are illustrated by an example.

The remainder of this work is structured as follows. Section 2 presents the problem formulation. In Section 3 we illustrate how to formulate the optimization problem by using only the states as optimization variables. Section 4 gives a detailed description of the proposed solution method. In Section 5 we illustrate the results. Finally, we provide a summary and outline future working directions.

2. PROBLEM FORMULATION

This section presents the class of considered systems and outlines the moving horizon estimation procedure.

2.1 Considered problem class

Time-invariant, discrete-time, linear systems of the form

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k, \\ y_k &= Cx_k + v_k. \end{aligned} \tag{1}$$

are considered, were $k \geq 0$ denotes the time, $x_k \in \mathbb{R}^n$ the state, $u_k \in \mathbb{R}^m$ the known input, $w_k \in \mathbb{R}^n$ the unknown process noise, $y_k \in \mathbb{R}^p$ the observed output and $v_k \in \mathbb{R}^p$ the unknown measurement noise. The matrices have appropriate dimensions and (A,C) is assumed to be detectable.

We assume that the state x_k and the measurement noise v_k are each constrained to polytopic, compact, convex sets: $x_k \in \mathbb{X}$ and $v_k \in \mathbb{V}$. Often, process noise w_k and measurement noise v_k are assumed to be (approximately) zero mean, Gaussian white noise with covariances given by Q and R, respectively, compare Rawlings and Mayne (2009). For the initial state x_0, an estimate $\tilde{x}_0$ with covariance Π_0 is available at $k = 0$. Note, while this provides for MHE somehow a relation to the stochastic optimal Kalman Filter, a pure deterministic setting is also possible. We assume that Q, R and Π_0 are positive definite.

Remark 1. (Neglection of bounds on the process noise w_k)
*Considering bounds only on the state and measurements is motivated by many practical applications: constraints on the state x can represent physical boundaries (*e.g. *concentrations, temperature, pressures, liquid level) and constraints on the measurements noise v_k can be obtained from the specification of the sensors (*e.g. *maximum error). In contrast, it is often more difficult to obtain bounds on the process noise w_k, except in a few special cases such as e.g. in (Rausch et al., 2014).*

2.2 Moving horizon estimation

To estimate the state x_k, $k > 0$ of system (1) based on the information available before the time instant k, one can use moving horizon estimation, see (Rao et al., 2001; Rawlings and Mayne, 2009) and the references therein for more details. The optimization based nature of MHE allows to take the bounds on x_k and v_k into account; in contrast to classical estimation techniques such as Kalman filtering, see Kailath et al. (2000); Rao et al. (2001); Rawlings and Mayne (2009).

The general idea in MHE to estimate x_k, c.f. Rawlings and Mayne (2009), is to solve at each time instant

$$\begin{aligned} \min_{\hat{\mathbf{v}},\hat{\mathbf{w}},\hat{\mathbf{x}}} & J(\hat{\mathbf{v}},\hat{\mathbf{w}},\hat{\mathbf{x}},\tilde{x}_s,\Pi_s) \\ \text{s.t. } & \hat{x}_{j+1} = A\hat{x}_j + Bu_j + w_j \\ & y_j = C\hat{x}_j + \hat{v}_j \\ & \hat{x}_i \in \mathbb{X},\ \hat{v}_j \in \mathbb{V}, \end{aligned} \tag{2}$$

where $s = \max(0, k-N)$ denotes the start of the estimation window, N is the (maximum) estimation window size, also called estimation horizon, $i = s,\dots,k$, $j = s,\dots,k-1$ and $\hat{\mathbf{x}} = \{\hat{x}_i\}$, $\hat{\mathbf{v}} = \{\hat{v}_j\}$, $\hat{\mathbf{w}} = \{\hat{w}_j\}$, $\mathbf{u} = \{u_j\}$ and $\mathbf{y} = \{y_j\}$. $\tilde{x}_s$, $s > 0$, is the prior estimate of x using the information available prior to $k = s$. The cost function J is given by

$$\begin{aligned} J(\hat{\mathbf{v}},\hat{\mathbf{w}},\hat{x}_s,\tilde{x}_s,\Pi_s) = & \frac{1}{2}\sum_{i=s}^{k-1} \|\hat{w}_i\|^2_{Q^{-1}} + \frac{1}{2}\sum_{i=s}^{k-1} \|\hat{v}_i\|^2_{R^{-1}} \\ & + \frac{1}{2}\|\tilde{x}_s - \hat{x}_s\|^2_{\Pi_s^{-1}}, \end{aligned} \tag{3}$$

where the choice of the so-called arrival cost matrix Π_s, which weights the influence of the estimate $\tilde{x}_s$ for $i > 0$, is discussed below. As an estimate for x_k at each time, the optimal value of $\hat{x}_k$ from (2) is used: $\tilde{x}_k = \hat{x}_k^\star$ [1].

The choice of the estimation windows size N is a trade-off between the maximum size of the optimization problem (2) and the estimation performance: using a smaller N reduces the computational demand, but can lead to deteriorated estimates.

Note that the optimization problem (2) is a convex, quadratic program, which needs to be solved at every time-instance, so an efficient solution is of key interest. Therefore, we present in the following section a tailored solution approach.

Remark 2. (Arrival cost matrix Π_i, choice of $\tilde{x}_i$)
The choice of the arrival cost matrix Π_i and the choice of $\tilde{x}_i$ are crucial for a good estimation performance. Incorrect choices can lead to inferior estimation and even an unstable estimation error dynamics, see (Rao et al., 2001; Rawlings and Mayne, 2009) for more details. We consider here only the simple approach using the prior state estimate $\tilde{x}_i$ as estimate of x_i computed at $k = i-1$ and to update Π_i using a Kalman filtering update

$$\Pi_{k+1} = A\Psi_k A^T + Q, \tag{4a}$$

$$\Psi_k = \Pi_k - \Pi_k C^T (C\Pi_k C^T + R)^{-1} C\Pi_k. \tag{4b}$$

Note that Π_k will converge to a unique fix point Π_∞, where the matrix Π_∞ as well as Π_i are positive definite, because (A,C) is detectable, $(A,W^{\frac{1}{2}})$ is stabilizable and V, Π_0 are positive definite, compare Kailath et al. (2000).

Remark 3. (Feasibility of MHE problem (2))
Since w_k is not constrained, the arising optimization problem is always feasible under the assumptions made, i.e. that the measurements are consistent with $\mathbb{V}$ and $\mathbb{X}$: for every y_k there

[1] Note that we consider here an estimate $\hat{x}_k$ using only information available prior to the time instance k: the so-called predicted or a priori estimate, compare Kailath et al. (2000). The proposed approach can be extended such that also y_k is used, i.e. to obtain the a posteriori (also called filtered) state estimate.

exists x_k and v_k such that $y_k = Cx_k + v_k$, $x_k \in \mathbb{X}$ and $v_k \in \mathbb{V}$. There is always a w_k such that (1) *is satisfied.*

3. REFORMULATION OF THE OPTIMIZATION PROBLEM

We first formulate the MHE optimization problem (2) such that the optimization is performed only over the state trajectory $\hat{\mathbf{x}}$, i.e. eliminating the noise sequences $\hat{\mathbf{w}}$ and $\hat{\mathbf{v}}$. This idea is inspired by similar ideas exploited in MPC, c.f. Mancuso and Kerrigan (2011), where the inputs are eliminated from the control variables to increase the speed of interior point methods.

First one can straightforwardly eliminate the measurement noise $\hat{\mathbf{v}}$ as optimization variable, see Haverbeke et al. (2009), by replacing $\|\hat{v}_i\|^2_{V^{-1}}$, $\hat{v}_i \in \mathbb{V}$ by $\|y_i - C\hat{x}_i\|^2_{V^{-1}}$ and $y_i - C\hat{x}_i \in \mathbb{V}$, respectively. The resulting optimization problem possess as optimization variables the state trajectory $\hat{\mathbf{x}}$ and the process noise sequence $\hat{\mathbf{w}}$. Furthermore, the optimization problem has a structure similar to so-called "sparse formulations" in MPC, see e.g. Mancuso and Kerrigan (2011); Wang and Boyd (2010).

Following the idea of Mancuso and Kerrigan (2011), we propose to additionally eliminate $\hat{\mathbf{w}}$ as optimization variables by using the state-space equation of the dynamics, i.e. replacing $\hat{w}_i$ in the cost function by $\hat{x}_{i+1} - A\hat{x}_i - Bu_i$.

Expressing the cost function (3) in terms of $\hat{\mathbf{x}}$ results in

$$\begin{aligned}\underline{J}(\hat{\mathbf{x}}, \tilde{x}_s, \Pi_s, \mathbf{u}, \mathbf{y}) =& \frac{1}{2}\|\tilde{x}_s - \hat{x}_s\|^2_{\Pi_s^{-1}} + \frac{1}{2}\sum_{i=s}^{k-1}\|y_i - C\hat{x}_i\|^2_{R^{-1}} \\ &+ \frac{1}{2}\sum_{i=s}^{k-1}\|\hat{x}_{i+1} - A\hat{x}_i - Bu_i\|^2_{Q^{-1}}. \end{aligned} \quad (5)$$

This allows to formulate the cost function in a compact way

$$\underline{J}(\hat{\mathbf{x}}, \tilde{x}_s, \Pi_s, \mathbf{u}, \mathbf{y}) = \frac{1}{2}\hat{\mathbf{x}}^T H \hat{\mathbf{x}} + \hat{\mathbf{x}}^T f(\tilde{x}_s, \mathbf{y}, \mathbf{u}) + g(\tilde{x}_s, \mathbf{y}, \mathbf{u}), \quad (6)$$

where H is a symmetric, positive-definite, block-tridiagonal matrix and $f(\tilde{x}_{k-N}, \mathbf{y}, \mathbf{u})$ is a vector that can be written as $f(\tilde{x}_s, \mathbf{y}, \mathbf{u}) = \tilde{f}(\tilde{x}_s, \mathbf{y}) + F\mathbf{u}$. H, F and f are given by

$$\tilde{f}(\tilde{x}_s, \mathbf{y}) = \begin{pmatrix} My_s - \Pi_s^{-1}\tilde{x}_s \\ My_{s+1} \\ \vdots \\ My_{k-1} \\ \mathbf{0} \end{pmatrix}, \quad H = \begin{pmatrix} U & G^T & 0 & \dots & 0 \\ G & D & G^T & \dots & 0 \\ 0 & G & D & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & Q^{-1} \end{pmatrix},$$

$$F = \begin{pmatrix} A^TQ^{-1}B & \dots & 0 \\ -Q^{-1}B & \dots & 0 \\ \vdots & \ddots & \vdots \\ \vdots & \ddots & A^TQ^{-1}B \\ 0 & 0 & -Q^{-1}B \end{pmatrix},$$

where $\mathbf{0}$ is the $\mathbb{R}^n$ zero vector and

$$\begin{aligned} U &= A^TQ^{-1}A + C^TR^{-1}C + \Pi_s^{-1}, & M &= -C^TR^{-1}, \\ D &= A^TQ^{-1}A + C^TR^{-1}C + Q^{-1}, & G &= -Q^{-1}A. \end{aligned}$$

Finally, problem (2) with $\hat{\mathbf{x}}$ as optimization variables becomes

$$\min_{\hat{\mathbf{x}} \in \mathbb{Z}(\mathbf{y})} \underline{J}(\hat{\mathbf{x}}, \tilde{x}_s, \Pi_s, \mathbf{u}, \mathbf{y}), \quad (7)$$

where $\mathbb{Z}(\mathbf{y})$ with $\mathbb{Y}(y) = \{x | y - Cx \in \mathbb{V}\}$ is given by

$$\mathbb{Z}(\mathbf{y}) = (\mathbb{X} \cap \mathbb{Y}(y_s)) \times \dots \times (\mathbb{X} \cap \mathbb{Y}(y_{k-1})) \times \mathbb{X}. \quad (8)$$

Observe that the optimization problem (7) has no equality constraints, all inequality constraints are formulated on single stages ($\mathbb{X} \cap \mathbb{Y}(y_i)$ depends only on $\hat{x}_i$ and y_i) and the matrix H is *time-varying* and has a block tridiagonal structure. These features provide the basis for the tailored algorithm presented in the following section.

4. FAST GRADIENT METHOD BASED MHE

To solve (2) we propose to use Nesterov's fast gradient method, see (Nesterov, 1983, 2004) for more details.

Nesterov's fast gradient method, sketched in Algorithm 1, is well suited for optimization problems of the form (7). Note that L and μ are upper and lower bounds on the eigenvalues of H: $L \geq \lambda_{\max}(H)$, $\mu \leq \lambda_{\min}(H)$. Furthermore, $\text{proj}(\xi, \mathbb{Z}(\mathbf{y}))$ is a Euclidean projection onto $\mathbb{Z}(\mathbf{y})$. After every instance the suboptimality of the current estimate is investigated. In detail, a bound e on $\underline{J}^\star - \underline{J}(\hat{\mathbf{x}}^i)$, i.e. the difference between the current solution $\underline{J}(\hat{\mathbf{x}}^i)$ and the optimal value $\underline{J}^\star$ is evaluated. The algorithm terminates if this bound is below a specified threshold or if a maximum number of iterations is reached.

Key points for a simple, yet efficient implementation are the stopping criterion, the projections and the computation of the largest/smallest eigenvalue of H as discussed in the following. We refer to Section 4.5 for the overall MHE algorithm.

Algorithm 1 Fast gradient algorithm

Require: H, f, L, $\mu > 0$, $\hat{\mathbf{x}}^0$, ε, $\mathbb{Z}(\mathbf{y})$, $i_{\max}$
  Set $\mathbf{z}^0 = \hat{\mathbf{x}}^0$, $e = \infty$, $i = 0$
  while $e > \varepsilon$ and $i \leq i_{\max}$ **do** ▷ Stopping criterion
    $i = i + 1$
    $\nabla\underline{J}(\mathbf{z}^{i-1}) = H\mathbf{z}^{i-1} + f$ ▷ Gradient computation
    $\xi = \mathbf{z}^{i-1} - \frac{1}{L}\nabla\underline{J}(\mathbf{z}^{i-1})$
    $\hat{\mathbf{x}}^i = \text{proj}(\xi, \mathbb{Z}(\mathbf{y}))$ ▷ Projection onto $\mathbb{Z}(\mathbf{y})$
    $\mathbf{z}^i = \hat{\mathbf{x}}^i + \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(\hat{\mathbf{x}}^i - \hat{\mathbf{x}}^{i-1})$
    Bound suboptimality e
  end while
  return $\hat{\mathbf{x}}^i$

4.1 Stopping criteria

In order to balance the accuracy of the solution with the computation time, we aim at using tailored stopping criteria to stop the fast gradient algorithm once a good enough solution has been found. We focus on two different types of criteria to determine when to stop. First we investigate an online criterion based on the results of Richter and Morari (2012, Section IV-B). Second we consider an offline criterion, which computes a worst case bound on the maximum number of iterations using results of Nesterov (1983, 2004); Richter et al. (2012).

Online stopping criterion: As online stopping criterion we use the results for the fast gradient method of Richter and Morari (2012, Section IV-B), which is based on a lower bound on the so-called gradient mapping. The main idea is to use a *lower bound* $\underline{J}_l^i$ on the optimal value $\underline{J}^*$, such that

$$\underline{J}(\hat{\mathbf{x}}^i) - \underline{J}^* \leq \underline{J}(\hat{\mathbf{x}}^i) - \underline{J}_l^i \leq \varepsilon, \quad (9)$$

where ε is the required tolerance. Additionally, one demands also that the sequence $\{\underline{J}_l^i\}_{i=0}^\infty$ has the property $\underline{J}_l^i \to \underline{J}^*$ as $i \to \infty$, which enables to achieve arbitrary small ε. Using properties

of gradient projection (Nesterov, 1983, 2004), the following criterion can be obtained (Richter and Morari, 2012):

$$\frac{1}{2}\left(\frac{1}{\mu}-\frac{1}{L}\right)\|L(\mathbf{z}^{i-1}-\mathbf{x}^{i})\|^2 \leq \varepsilon, \tag{10}$$

where $\mathbf{x}$ and $\mathbf{z}$ are as in Algorithm 1.

Offline stopping criterion / Bound on the worst case number of iterations: As alternative to an online stopping criterion one can compute the number of iterations offline, which guarantees in the worst case a certain suboptimality of the solution, by utilizing results adapted from the MPC case (Richter et al., 2012). This requires $\hat{\mathbf{x}}^0 = \text{proj}(\hat{\mathbf{x}}^\alpha - \frac{1}{L}(H\hat{\mathbf{x}}^\alpha + f), \mathbb{Z}(\mathbf{y}))$ as starting point, where $\mathbf{x}^\alpha$ is an initial guess.

Using the results of Richter et al. (2012), we obtain the following upper bound on the iterations number

$$i^{\max} = \min\left\{\left\lceil\frac{\ln 2\varepsilon - \ln Ld^2}{\ln\left(1-\sqrt{\frac{\mu}{L}}\right)}\right\rceil, \left\lceil\sqrt{\frac{2Ld^2}{\varepsilon}}-2\right\rceil\right\}, \tag{11}$$

which ensures that $\underline{J}(\hat{\mathbf{x}}^i) - \underline{J}^\star \leq \varepsilon$, where

$$d(\hat{\mathbf{x}}^\alpha) = \max_{\hat{\mathbf{x}}\in\mathbb{Z}(\mathbf{y})} \|\hat{\mathbf{x}} - \hat{\mathbf{x}}^\alpha\|. \tag{12}$$

Note that the bound d depends on $\mathbf{y}$. We can compute such an upper bound $d^{\max}$ on d, by using

$$d^{\max}(\hat{\mathbf{x}}^\alpha) = \max_{\hat{\mathbf{x}}\in\mathbb{X}\times\mathbb{X}\times\ldots\times\mathbb{X}} \|\hat{\mathbf{x}} - \hat{\mathbf{x}}^\alpha\|^2. \tag{13}$$

Remark 4. (Choice of $\hat{\mathbf{x}}^\alpha$)
There are two alternatives for the choice of the initial guess $\hat{\mathbf{x}}^\alpha$. First we can use cold-starting, i.e. always use the same initial guess $\hat{\mathbf{x}}^\alpha$. If we choose $\hat{\mathbf{x}}^\alpha$ such that $d^{\max}$ is minimized, then this reduces the upper bound on the required number of iterations (11) *and thus the worst case complexity, (Richter et al., 2012).*

A second alternative is to use so-called warm starting*, i.e. to choose $\hat{\mathbf{x}}^\alpha$ based on the solution at the time step before in order to be closer to the next solution. This reduces usually the number of iterations required until the stopping criterion* (10) *is satisfied. Unfortunately, it is challenging to obtain a bound on the number of required iteration in the worst-case. $\mathbf{x}^\alpha$.*

4.2 Euclidean projections

The fast gradient algorithm (Algorithm 1) requires in each step a Euclidean projection. This projection is defined by

$$\text{proj}(\xi, \mathbb{Z}(\mathbf{y})) = \arg\min_{\zeta\in\mathbb{Z}(\mathbf{y})} \|\zeta - \xi\|_2^2. \tag{14}$$

Since $\mathbb{Z}(\mathbf{y})$ is a polytopic set, the projection requires to solve a convex, quadratic program. Fortunately, $\mathbb{Z}(\mathbf{y})$ is separable, so that this projection can be done stage-wise:

$$\text{proj}(\xi, \mathbb{Z}(\mathbf{y})) = \begin{pmatrix} \text{proj}(\xi_s, \mathbb{X}\cap\mathbb{Y}(y_s)) \\ \vdots \\ \text{proj}(\xi_{k-1}, \mathbb{X}\cap\mathbb{Y}(y_{k-1})) \\ \text{proj}(\xi_k, \mathbb{X}) \end{pmatrix}. \tag{15}$$

This means we need to solve in general up to $N+1$ quadratic programs for the projection, each with n decision variables. However, simple, explicit representation of the projection $\text{proj}(\xi_i, \mathbb{X}\cap\mathbb{Y}(y_i))$ are possible for specific types of sets.

For example if $\mathbb{X}\cap\mathbb{Y}(y_i)$ is a box, then the projection is rather simple, c.f. Nesterov (2004). Note that $\mathbb{X}\cap\mathbb{Y}(y_i)$ is a box, if $\mathbb{X}$ as well as $\mathbb{V}$ are boxes and C contains only one entry per row (single states are measured). In our experience such setups can often be found in chemical engineering applications.

4.3 Computational effort of the fast gradient method

The amount of calculation required for the fast gradient algorithm (Algorithm 1) is dominated by the computation of the gradient and/or the effort for the Euclidean projections, compare Section 4.2.

For the gradient computation, which requires a matrix-vector multiplication and a vector addition, one can exploit the special structure of the H matrix to avoid unnecessary computations. Clearly, if multiple computational entities are available, one can easily parallelize the projections, which can be done stage-wise, as well as in form of the gradient computation, due to the special structure of H. This can significantly reduce the computational time.

With respect to the memory demand we need to store the vectors x^i, x^{i-1}, z^i, f, the set $\mathbb{Z}(\mathbf{y})$ and the matrices U, G, D, Q^{-1} to express the matrix H.

Note that for the overall MHE algorithm, we need additional computations to update f, which requires additionally to store u, y, F, Π_s and the system data (A, B, C, Q, R). Finally, one needs to also compute L and μ, which requires some additional computation and also increases the memory demand as outlined in the following. The main difference to MPC is the necessity to adapt L and μ at all time steps, since H is time-varying.

4.4 Efficient determining bounds on the eigenvalues

The fast gradient method requires bounds on the maximum and minimum eigenvalues of the matrix H, denoted by L and $\mu > 0$ on them respectively.

Unfortunately, the matrix H is here, in contrast to MPC, in general time-varying as (a) its size grows for the first N steps and (b) the arrival cost Π_k is time-varying, unless Π_0 is equal to the steady state value, see e.g. Rawlings and Mayne (2009); Richter et al. (2012); Wang and Boyd (2010). Therefore, one needs to compute these bounds at each time step.

In principle, one can easily obtain an upper bound on the eigenvalues: $L = \|H\|_1$. In contrast, finding a lower bound $\mu > 0$ is not that easy. For example $\mu = (\|H^{-1}\|_1)^{-1}$ requires inverting the matrix H, a rather large computational effort. Additionally, note that using loose bounds on the eigenvalues of H can decrease the convergence speed. Therefore, we propose to compute $\lambda_{\max}(H)$ and $\lambda_{\min}(H)$ online at each step.

The inverse iteration: To tackle the challenge of computing $\lambda_{\max}(H)$ and $\lambda_{\min}(H)$ we propose to use an iterative method called inverse iteration. This method allows one to exploit the structure of the matrix H. In the following we shortly review the inverse iteration, for more details we refer to Golub and Van Loan (2012). Basically, the inverse iteration computes an eigenvalue $\overline{\lambda}_i$ of the matrix H iteratively by

$$q_i = \frac{r_{i-1}}{\|r_{i-1}\|_2} \tag{16a}$$

$$r_i = (H-\theta I)^{-1} q_i \tag{16b}$$

$$\overline{\lambda}_i = \frac{r_i^T H r_i}{r_i^T r_i}, \tag{16c}$$

where $\theta \in \mathbb{R}$ is the so-called shift and $r_0 \neq 0$ is a starting vector. The choice of the shift θ determines the convergence. In detail, it is known that $\overline{\lambda}_i$ converges to the eigenvalue, which is closest to θ. This enables to compute the extreme eigenvalues of H: for the choice $\theta = \|H\|_1 > \lambda_{\max}(H)$ ($\theta = 0 < \lambda_{\min}(H)$) we have $\overline{\lambda}_i \to \lambda_{\max}(H)$ ($\overline{\lambda}_i \to \lambda_{\min}(H)$).

Implementation of the inverse iteration: In order to reduce the computational burden, we aim to avoid the explicit computation of the inverse of $M = H - \theta I$. Note that $(H-\theta I)^{-1}$ is symmetric and constant within the inverse iteration methods, therefore we propose to utilize Cholesky factorizations to solve efficiently the system $Mr_i = q_i$.

Since, for $\theta = 0$, M is positive definite, we can proceed straightforwardly, compare Golub and Van Loan (2012): compute the Cholesky factorization $KK^T = M$ at the beginning (K is a lower triangular matrix). In a second step solve $Mr_i = q_i$ at every iteration of the inverse iteration via Cholesky substitutions:

$$Ks_i = q_i, \; K^T r_i = s_i. \tag{17}$$

For the case $\theta = \|H\|_1$, $M = H - \|H\|_1 I$ is negative definite, but $(-M)r_i = -q_i$ can be solved similarly as above.

Computational effort of inverse iterations: The computational effort in the proposed approach is governed by the Cholesky factorizations of H and $\|H\|_1 I - H$, which have to be performed once per execution of the inverse iteration, as well as the computation of Hr_i (or $(\|H\|_1 I - H)r_i$) and the forward/backward substitutions in every iteration of the method. Clearly, Hr_i can be efficiently computed similarly as above by exploiting the problem structure. Note that also for the Cholesky factorization and substitutions the block tridiagonal structure of H can be utilized, see e.g. Wang and Boyd (2010), which can significantly reduce the computational load and the memory demand for large N.

Finally, note that the H matrix depends only on the arrival cost matrix Π_k and the time instance k, i.e. not on the actual measurements or inputs. This allows to compute the eigenvalues $\lambda_{Max}(H)$ and $\lambda_{Min}(H)$ before the measurement is known. So one can do these computations in advance and once y_k, u_k are available, one can start with the MHE reducing the time delay of the estimation procedure.

4.5 Overall MHE algorithm

The proposed overall MHE algorithm is given by:

1) Update H, f, Π_k (see Sections 2.2, 3).
2) Compute L, μ with inverse iteration (see Section 4.4).
3) Wait until y_{k-1} and u_{k-1} are available.
4) Solve (7) with Nesterov's gradient method (Algorithm 1).
5) Return $\hat{x}_k^i$ and set $\tilde{x}_{k+1} = \hat{x}_k^i$.

Steps 1 and 2 can be done before y_{k-1} and u_{k-1} become available, because they are independent of these values.

5. SIMULATION RESULTS

We consider a chemical reaction system formed by two continuous stirred-tank reactors (CSTRs) followed by a non-adiabatic flash separator (see Figure 1) taken from Venkat et al. (2006). The overall reaction consists of:

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C,$$

where B is the product and C an unwanted side product. These reaction take place in both reactors. The product of the second CSTR is sent to the flash for separating the excess A which has higher relative volatility than B and C. The vapor phase, which is rich in A, is partially purged and the remaining part is condensed back to the first CSTR. For details regarding the plant scheme we refer to Venkat et al. (2006, Example 8.2).

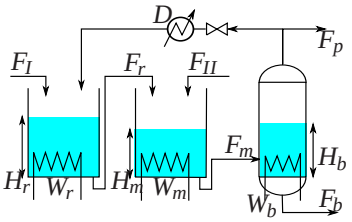

Fig. 1. Example system: Two reactors followed by a flash drum.

The system is *nonlinear* and *stable*, with 12 states (liquid levels, temperature, concentration of A and B in each subsystem) and 6 control inputs (flows F_I, F_{II}, D and heat exchangers W_i). We assume that all liquid levels and all temperatures are measured, i.e. we have 6 measurements. To obtain a linear model we linearise the system around the steady state and discretize it with a sampling time of 0.1s.

We consider box constraints on the states as well as on the measurement noise

$$\mathbb{X} = \{x \in \mathbb{R}^n | x^l \leq x_k \leq x^u\},$$
$$\mathbb{Y}_k = \{x \in \mathbb{R}^n | v^l \leq y_k - Cx_k \leq v^u\},$$

in which x^l and x^u are lower and upper bounds on the states (physical bounds *e.g.* minimum and maximum height or mass fraction) and v^l/v^u are lower/upper bound on the measurement noise, i.e. on the sensors accuracy.

Fig. 2 reports the result for a characteristic state the MHE result utilizing a suboptimality threshold of $\varepsilon = 10^{-4}$ and a window size of $N = 20$ using the proposed algorithm. The process and measurement noise have been simulated as uniform random noise.

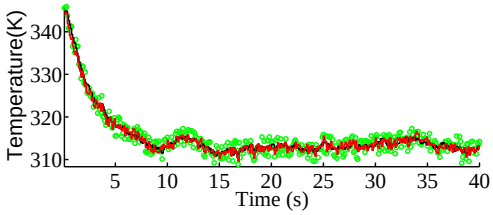

Fig. 2. Temperature in flash separator. (Red: estimates, green: measurements, black: real values)

To illustrate the performance we used an Intel Core i7 - 4770 CPU with a clock frequency of 3.4 GHz and a simple prototype implementation of the proposed approach. In Fig. 3 we illustrate for different horizon lengths along a simulation of 500 time steps the total computation time spent in the proposed approach and the time required by the fast gradient method only. As mentioned in Section 4.5 one can do the computation related to the inverse iteration/Kalman filtering formulas before the measurements and inputs at k are available.

For comparison, the computation times using the Matlab solver *quadprog* requires 1.95s for $N = 5$ using the active-set method

and 18.4s for $N = 50$ using the structure-exploiting interior point method (we have chosen the fastest method from *quadprog* for each case). So for this example the simple implementation of the proposed approach is around 3.5 times faster, if the computation time of the inverse iteration are included, and between 5 (for $N = 50$) to 7 (for $N = 5$) times faster if the time spent in the inverse iteration and the Kalman filtering formulas are excluded.

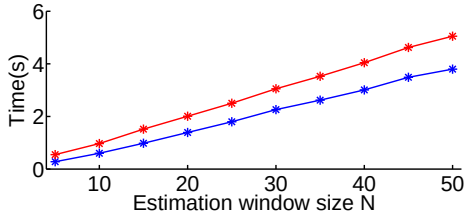

Fig. 3. Total computation times for 500 time steps. Blue: time of fast gradient method only. Red: time of overall algorithm.

6. SUMMARY AND FUTURE WORKING DIRECTIONS

We investigated the application of Nesterov's fast gradient method (Nesterov, 1983, 2004) to moving horizon estimation for linear system. In particular, we formulated the problem such that only the states $\{\hat{x}_i\}$ are optimization variables and proposed to use the inverse iteration to compute the required extreme eigenvalues of the time-varying Hessian matrix online. We discussed the implementation of the overall approach. A simple example illustrated the performance of the proposed algorithm.

Future works will focus on a more detailed analysis of the proposed approach, a comparison of the proposed approach with existing methods and extensions to other classes of moving horizon estimation. Moreover, we want to investigate the influence of the inaccurate solution onto the estimation performance and the stability of the estimation error. Additionally, we plan to extend the approach to nonlinear moving horizon estimation.

REFERENCES

Abrol, S. and Edgar, T.F. (2011). A fast and versatile technique for constrained state estimation. *Journal of Process Control*, 21(3), 343–350.

Darby, M.L. and Nikolaou, M. (2007). A parametric programming approach to moving-horizon state estimation. *Automatica*, 43(5), 885 – 891.

Diehl, M., Findeisen, R., Allgöwer, F., Bock, H., and Schlöder, J. (2005). Nominal stability of real-time iteration scheme for nonlinear model predictive control. *IEE Proceedings - Control Theory and Applications*, 152(3), 296–308.

Faulwasser, T., Lens, D., and Kellett, C.M. (2014). Predictive control for longitudinal beam dynamics in heavy ion synchrotrons. In *Proc. IEEE Conf. on Control Applications*, 1988–1995.

Golub, G.H. and Van Loan, C.F. (2012). *Matrix Computations*, volume 3. Johns Hopkins University Press.

Haverbeke, N., Diehl, M., and De Moor, B. (2009). A structure exploiting interior-point method for moving horizon estimation. In *Proc. IEEE Conf. on Decision & Control and Chinese Control Conf.*., 1273–1278.

Jang, H., Lee, J.H., Braatz, R.D., and Kim, K.K.K. (2014). Fast moving horizon estimation for a two-dimensional distributed parameter system. *Computers & Chemical Engineering*, 63, 159–172.

Jerez, J., Goulart, P., Richter, S., Constantinides, G., Kerrigan, E., and Morari, M. (2014). Embedded Online Optimization for Model Predictive Control at Megahertz Rates. *IEEE Transactions on Automatic Control*, 59(12), 3238–3251.

Kailath, T., Sayed, A.H., and Hassibi, B. (2000). *Linear Estimation*. Prentice Hall, Upper Saddle River.

Kögel, M. and Findeisen, R. (2011). A fast gradient method for embedded linear predictive control. In *Proc. IFAC World Congress*, 1362–1367.

Kraus, T., Kühl, P., Wirsching, L., Bock, H., and Diehl, M. (2006). A Moving Horizon State Estimation algorithm applied to the Tennessee Eastman benchmark process. In *Proc. IEEE Robotics & Automation Society Conf. on Multisensor Fusion and Integration for Intelligent Systems*.

Mancuso, G.M. and Kerrigan, E.C. (2011). Solving constrained LQR problems by eliminating the inputs from the QP. In *Proc. IEEE Conf. on Decision & Control and European Control Conf.*, 507–512.

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2), 372–376.

Nesterov, Y. (2004). *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Pubishers, Boston.

Patrinos, P. and Bemporad, A. (2014). An accelerated dual gradient-projection algorithm for embedded linear model predictive control. *IEEE Transactions on Automatic Control*, 59(1), 18–33.

Rao, C.V., Rawlings, J.B., and Lee, J.H. (2001). Constrained linear state estimation - a moving horizon approach. *Automatica*, 37(10), 1619–1628.

Rausch, M., Klein, R., Streif, S., Pankiewitz, C., and Findeisen, R. (2014). Set-based state of charge estimation for lithium-ion batteries. In *Proc. American Control Conf.*, 1566–1571.

Rawlings, J.B. and Mayne, D.Q. (2009). *Model Predictive Control: Theory and Design*. Nob Hill Pubishing.

Richter, S. and Morari, M. (2012). Stopping Criteria for First-Order Methods. Technical report, ETH Zurich, Switzerland.

Richter, S., Jones, C.N., and Morari, M. (2012). Computational Complexity Certification for Real-Time MPC With Input Constraints Based on the Fast Gradient Method. *IEEE Transactions on Automatic Control*, 57(6), 1391–1403.

Richter, S., Mariéthoz, S., and Morari, M. (2010). High-speed online MPC based on a fast gradient method applied to power converter control. In *Proc. American Control Conf.*, 4737–4743.

Venkat, A.N., Rawlings, J.B., and Wright, S.J. (2006). Stability and optimality of distributed, linear model predictive control. Part I: State feedback. Technical Report 2006-03, Texas-Wisconsin Modeling and Control Consortium.

Wang, Y. and Boyd, S. (2010). Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, 18(2), 267–278.

Zavala, V.M., Laird, C.D., and Biegler, L.T. (2008). A fast moving horizon estimation algorithm based on nonlinear programming sensitivity. *Journal of Process Control*, 18(9), 876–884.

Zometa, P., Kögel, M., Faulwasser, T., and Findeisen, R. (2012). Implementation aspects of model predictive control for embedded systems. In *Proc. American Control Conf.*, 1205–1210.