Computers and Chemical Engineering 63 (2014) 159–172

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

Fast moving horizon estimation for a two-dimensional distributed parameter system☆

Hong Jang[a], Jay H. Lee[a,*], Richard D. Braatz[b], Kwang-Ki K. Kim[b,c]

[a] Department of Biomolecular and Chemical Engineering, Korea Advanced Institute of Science and Technology, 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Republic of Korea
[b] Department of Chemical Engineering, Massachusetts Institute of Technology, MA 02139, United States
[c] School of Electrical and Computer Engineering, Georgia Institute of Technology, GA 30332, United States

ARTICLE INFO

Article history:
Received 7 May 2013
Received in revised form 4 December 2013
Accepted 18 December 2013
Available online 28 December 2013

Keywords:
Moving horizon estimation
Kalman filter
Ellipsoid constraint
Singular value decomposition
Penalty method
Partial differential equation

ABSTRACT

Partial differential equations (PDEs) pose a challenge for control engineers, both in terms of theory and computational requirements. PDEs are usually approximated by ordinary differential equations or difference equations via the finite difference method, resulting in a high-dimensional state-space system. The obtained system matrix is oftentimes symmetric, which allows this high-dimensional system to be decomposed into a set of single-dimensional systems using its singular value decomposition. Any linear constraints in the original problem can also be simplified by replacing it with an ellipsoidal constraint. Based on this, speedup of the moving horizon estimation is achieved by employing an analytical solution obtained by augmenting the ellipsoidal constraint into the objective function as a penalty weighted by a decreasing scaling parameter. The approximated penalty method algorithm allows for efficient parallel computation for sub-problems. The proposed algorithm is demonstrated for a two-dimensional diffusion problem where the concentration field is estimated using distributed sensors.

1. Introduction

Distributed parameter systems (DPS) are found in a variety of engineering applications in aerospace, materials, chemistry, and biology. The state and output responses of these systems are functions of spatial and temporal variables and thus described by a system of partial differential equations (PDEs). In comparison to ordinary differential equations (ODEs) or differential-algebraic equations (DAEs), PDEs describe physical systems more accurately but are more challenging to handle both theoretically and computationally.

For the purpose of controller design for DPS, infinite dimension PDEs are usually approximated by finite dimension ODEs or difference equations via the finite difference method (Morton & Mayers, 1995). The advantage of this approximation is that many control design methods including model predictive control (MPC) are directly applicable to the approximate finite-dimensional system. However, to obtain a reasonable numerical solution, the approximation often requires a very large number of variables, resulting in a system of ODEs of very high dimension. On the other hand, controller design of a DPS by directly using the PDE often requires mathematical knowledge (such as infinite-dimensional operator theory (Curtain & Zwart, 1995) or non-harmonic Fourier series (Russell, 1967)) that is unfamiliar to most control engineers.

State estimation is an important element of modern control applications. The Kalman filter (KF) is the current standard for state estimation of a linear system (Kalman, 1960; Kalman & Bucy, 1961). However, the KF in its bare form cannot incorporate constraints, which may come from nature of the physics or process knowledge. Quadratic programming (QP)-based moving horizon estimation (MHE) has been suggested as a practical way to incorporate inequality constraints into state estimation (Muske, Rawlings, & Lee, 1993; Rao, Rawlings, & Lee, 2001; Robertson, Lee, & Rawlings, 1996). A drawback of this optimization-based approach is the substantially higher computational cost when compared to explicit estimators such as the KF. This drawback may limit the size of the problem to which the MHE method can be applied.

☆ A preliminary version of the manuscript was published as H. Jang, K.-K. K. Kim, J. H. Lee, and R. D. Braatz, Fast moving horizon estimation for a distributed parameter system, In 12th International Conference on Control, Automation and Systems, pp. 533–538, 2012, Jeju Island, Korea.

* Corresponding author. Tel.: +82 42 350 3926; fax: +82 42 350 3910.
E-mail addresses: jayhlee@kaist.ac.kr, eva@kt.dtu.dk (J.H. Lee).

0098-1354/$ – see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.compchemeng.2013.12.005

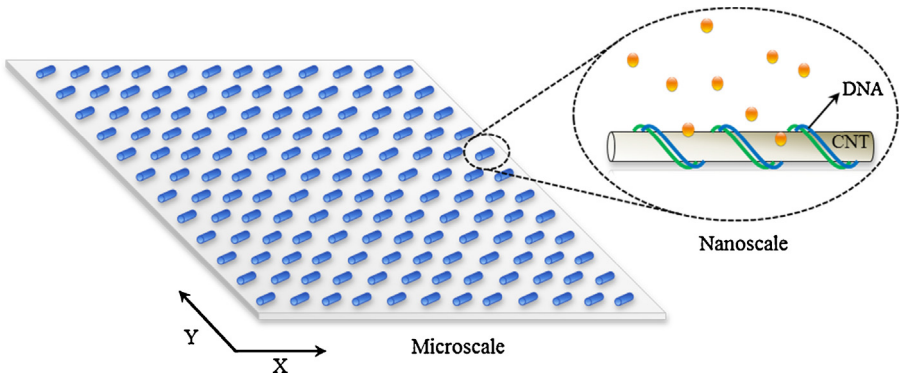

Fig. 1. Carbon nanotube-based sensor arrays system distributed uniformly on a two-dimensional spatial field (left) and example sensing mechanism on the surface of each local sensor (Boghossian et al., 2011; Jin et al., 2010; Zhang et al., 2010) (right).

In this paper, a *fast* MHE method is proposed for the estimation of states in DPS. The method is tailored for a very high-dimensional state-space system resulting from the application of the finite difference method to a system of PDEs. The high-dimensional system is decoupled into individual single-dimensional systems through singular value decomposition (SVD). The original polytopic inequality constraints (typically enforced in MHE) are approximated by an ellipsoidal constraint that is handled by a penalty method algorithm for the decoupled system. The computational savings resulting from the approximate method are examined using a system described by the two-dimensional diffusion equation.

A prospective application of this method is a system equipped with a nanomaterial-based sensor array for estimating a 2D spatial concentration field. Carbon nanotube (CNT)-based sensors have been experimentally demonstrated for estimating local changes in the concentrations of target molecules in a liquid solution, even near the surface of living cells (Boghossian et al., 2011; Jin et al., 2010; Zhang et al., 2010). By employing these sensors placed in a uniform array of some 2D spatial domain, the concentration fields of target signal molecules can be estimated using the MHE method described in this paper (Fig. 1). Real-time estimation of the concentration fields from such a sensor array would be useful for a control system designed to achieve a specified 2D spatial control objective. For example, the real-time state estimation could be fed into an optimal control system that injects special molecules into the process to achieve a specific temporal and spatial concentration patterns, which induces a desired stem cell differentiation (Kishida, Pack, & Braatz, 2010).

In Section 2, a two-dimensional diffusion equation is introduced as a main representative governing equation describing the target sensing system. In Section 3, decentralized formulations of conventional KF and MHE are suggested for decentralized and independent subsystems of the target system without constraints. Then, as a new result, constrained decentralized MHE is proposed by using an ellipsoidal constraint that approximates a system of linear inequalities in Section 4 and its performance is compared with the original MHE in Section 5. Some conclusions are given in Section 6.

2. Partial difference equation

The partial differential equation for 2D diffusion is given by

$$\frac{\partial C(X,Y,t)}{\partial t} = D\left(\frac{\partial^2 C(X,Y,t)}{\partial X^2} + \frac{\partial^2 C(X,Y,t)}{\partial Y^2}\right) \tag{1}$$

where $C(X, Y, t)$ is the concentration of a substance of interest at position (X, Y) at time t and D is a known diffusion coefficient.

For a uniform mesh of size $(\Delta X, \Delta Y)$ and a constant discretization time $\Delta t = t_{k+1} - t_k$, the partial differential Eq. (1) can be converted into the *difference equation*,

$$\frac{C_{i,j}^{k+1} - C_{i,j}^{k}}{\Delta t} = D\left(\frac{C_{i+1,j}^{k} - 2C_{i,j}^{k} + C_{i-1,j}^{k}}{(\Delta X)^2} + \frac{C_{i,j+1}^{k} - 2C_{i,j}^{k} + C_{i,j-1}^{k}}{(\Delta Y)^2}\right) \tag{2}$$

$$C(X_i, Y_j, t_k) \equiv C_{i,j}^{k}; \quad i = 1, \ldots, n_X; \quad j = 1, \ldots, n_Y; \quad k = 0, \ldots, n_T \tag{3}$$

where a first-order forward finite difference equation was used for the time derivative and a second-order finite difference expression was used for the spatial derivative. Time can be marched forward by solving for $C_{i,j}^{k+1}$ as

$$C_{i,j}^{k+1} = C_{i,j}^{k} + s_X(C_{i+1,j}^{k} + C_{i-1,j}^{k} - 2C_{i,j}^{k}) + s_Y(C_{i,j+1}^{k} + C_{i,j-1}^{k} - 2C_{i,j}^{k}) \tag{4}$$

$$s_X = D\frac{\Delta t}{(\Delta X)^2}; s_Y = D\frac{\Delta t}{(\Delta Y)^2} \tag{5}$$

Numerical stability of the difference equation depends on the value of the scalars s_X and s_Y, which can be chosen to obtain a stable numerical solution of required accuracy. Fourier-von Neumann stability analysis indicates that the finite difference method is numerically stable provided that (Haberman, 2004)

$$s_X + s_Y \leq \frac{1}{2} \tag{6}$$

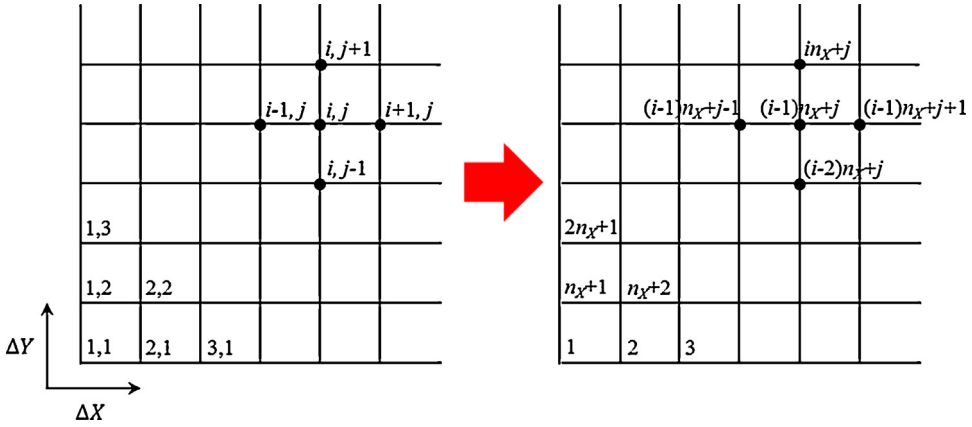

Fig. 2. Sequential node numbering on a two-dimensional mesh field.

By sequentially numbering the nodes at mesh points as shown in Fig. 2, the discrete-time state-space equation for the system (4) is

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) \tag{7}$$

$$\boldsymbol{x}(k) = \begin{bmatrix} C^k_{1,1} \\ C^k_{2,1} \\ \vdots \\ C^k_{i,j} \\ C^k_{i+1,j} \\ \vdots \\ C^k_{n_X-1,n_Y} \\ C_{n_X,n_Y} \end{bmatrix} \tag{8}$$

$$\boldsymbol{A}_1 = \begin{bmatrix} 1-2s_X-2s_Y & s_X & 0 & \cdots & \cdots & 0 \\ s_X & 1-2s_X-2s_Y & s_X & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & s_X & 0 \\ \vdots & \ddots & \ddots & s_X & 1-2s_X-2s_Y & s_X \\ 0 & \cdots & \cdots & 0 & s_X & 1-2s_X-2s_Y \end{bmatrix} \tag{9}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} s_Y & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & s_Y \end{bmatrix} \tag{10}$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{A}_2 & \boldsymbol{0} & \cdots & \cdots & \boldsymbol{0} \\ \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_2 & \ddots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{A}_2 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \boldsymbol{A}_2 & \boldsymbol{0} \\ \vdots & \ddots & \ddots & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_2 \\ \boldsymbol{0} & \cdots & \cdots & \boldsymbol{0} & \boldsymbol{A}_2 & \boldsymbol{A}_1 \end{bmatrix} \tag{11}$$

where $\mathbf{0}$ refers to the null matrix whose entries are all zeros with a compatible dimension. The system matrix $\boldsymbol{A}$ has $n_X n_Y \times n_X n_Y$ dimension and a symmetric structure.

3. Decoupled state estimation

Gaussian white noise terms can be added to Eq. (7) to produce the standard discrete-time stochastic state-space system description:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{\varepsilon}_1(k) \tag{12}$$

$$\boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{\varepsilon}_2(k) \tag{13}$$

This system is linear and time invariant (LTI) with constant system matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ and $\boldsymbol{\varepsilon}_1(k)$ and $\boldsymbol{\varepsilon}_2(k)$ are independent process and observation noises having Gaussian white distribution with covariance matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$, respectively. Throughout this paper, $\boldsymbol{C}$ is assumed to be an identity matrix $\boldsymbol{I}$, where individual sensors are placed at all nodes of the discretized 2D mesh to detect the local concentrations of target molecules. This assumption requires that the sensor array has a very fine spatial resolution or the spatial discretization used in the PDE is limited by the sensor array's spatial resolution. However, nano-sensors have been developed in recent years that enable such high spatial resolutions when used in a sensor array (Boghossian et al., 2011; Jin et al., 2010; Zhang et al., 2010).

The system matrix (11) resulting from the discretization has a symmetric structure. A symmetric matrix can be converted into a diagonal matrix via orthogonal transformation, which is an approach that has been used in fast MPC for large-scale systems (VanAntwerp & Braatz, 2000a). By using the *singular value decomposition* (SVD), which is equivalent to the Jordan decomposition in this symmetric case, the matrix $\boldsymbol{A}$ can be decomposed into three matrices:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T; \quad \boldsymbol{U}^T = \boldsymbol{U}^{-1} \tag{14}$$

where $\boldsymbol{U}$ is an orthonormal matrix. This equation can be inserted into Eqs. (12) and (13) to produce the transformed state-space model,

$$\tilde{\boldsymbol{x}}(k+1) = \boldsymbol{\Sigma}\tilde{\boldsymbol{x}}(k) + \tilde{\boldsymbol{\varepsilon}}_1(k); \quad \tilde{\boldsymbol{x}}(k) = \boldsymbol{U}^T\boldsymbol{x}(k), \quad \tilde{\boldsymbol{\varepsilon}}_1(k) = \boldsymbol{U}^T\boldsymbol{\varepsilon}_1(k) \tag{15}$$

$$\tilde{\boldsymbol{y}}(k) = \tilde{\boldsymbol{x}}(k) + \tilde{\boldsymbol{\varepsilon}}_2(k); \quad \tilde{\boldsymbol{y}}(k) = \boldsymbol{U}^T\boldsymbol{y}(k), \quad \tilde{\boldsymbol{\varepsilon}}_2(k) = \boldsymbol{U}^T\boldsymbol{\varepsilon}_2(k) \tag{16}$$

Since $\boldsymbol{\Sigma}$ is diagonal, the state vector can be separated into each element:

$$\tilde{x}_i(k+1) = \Sigma_{i,i}\tilde{x}_i(k) + \tilde{\varepsilon}_{1,i}(k) \tag{17}$$

$$\tilde{y}_i(k) = \tilde{x}_i(k) + \tilde{\varepsilon}_{2,i}(k) \tag{18}$$

A Kalman filter (KF) or a moving horizon estimator (MHE) can be developed for each individual state of the decoupled state space system.

3.1. Decoupled Kalman filter

The discrete-time linear Kalman filter for the state-space system of Eqs. (12) and (13) is

$$\hat{\boldsymbol{x}}(k|k-1) = \boldsymbol{A}\hat{\boldsymbol{x}}(k-1|k-1) \tag{19}$$

$$\hat{\boldsymbol{x}}(k|k) = \hat{\boldsymbol{x}}(k|k-1) + \boldsymbol{K}(k)(\boldsymbol{y}(k) - \hat{\boldsymbol{x}}(k|k-1)) \tag{20}$$

Eq. (19) is a simple propagation of the model state to next time step according to the model and is called *model forwarding*. Eq. (20) corrects the forwarded estimate based on the term called *innovation*, which is the difference between the actual measurement of the output and its predicted value from the current state estimate. This step is called the *measurement update*. The optimal gain matrix, $\boldsymbol{K}(k)$, and covariance matrix, $\boldsymbol{P}(k)$, are calculated according to

$$\boldsymbol{K}(k) = (\boldsymbol{A}\boldsymbol{P}(k-1)\boldsymbol{A}^T + \boldsymbol{Q})(\boldsymbol{R} + \boldsymbol{A}\boldsymbol{P}(k-1)\boldsymbol{A}^T + \boldsymbol{Q})^{-1} \tag{21}$$

$$\boldsymbol{P}(k) = (\boldsymbol{I} - \boldsymbol{K}(k))(\boldsymbol{A}\boldsymbol{P}(k-1)\boldsymbol{A}^T + \boldsymbol{Q}) \tag{22}$$

For the decoupled state-space system, the Kalman filter also becomes decoupled:

$$\hat{\tilde{x}}_i(k|k-1) = \Sigma_{i,i}\hat{\tilde{x}}_i(k-1|k-1) \tag{23}$$

$$\hat{\tilde{x}}_i(k|k) = \hat{\tilde{x}}_i(k|k-1) + \tilde{K}_{i,i}(k)(\tilde{y}_i(k) - \hat{\tilde{x}}_i(k|k-1)) \tag{24}$$

The individual Kalman gain, $\tilde{K}_{i,i}(k)$ and covariance, $\tilde{P}_{i,i}(k)$, can also be updated by

$$\tilde{K}_{i,i}(k) = \frac{\Sigma_{i,i}^2\tilde{P}_{i,i}(k-1) + \sigma_1}{\sigma_2 + \Sigma_{i,i}^2\tilde{P}_{i,i}(k-1) + \sigma_1} \tag{25}$$

$$\tilde{P}_{i,i}(k) = (1 - \tilde{K}_{i,i}(k))(\Sigma_{i,i}^2\tilde{P}_{i,i}(k-1) + \sigma_1) \tag{26}$$

where σ_1 and σ_2 are the scalar multipliers in $\boldsymbol{Q}$ and $\boldsymbol{R}$, respectively, and represent the variances of the respective noise terms.

3.2. Decoupled moving horizon estimator

The basic strategy of MHE for an LTI system is to reformulate the estimation problem as a quadratic program (QP) on a moving fixed-size estimation window (Rao et al., 2001). A finite-size estimation window (of size n) is used to bind the size of the quadratic program.

The main idea is that the objective function for the batch least-squares estimation can be rearranged by breaking the time interval into two pieces: $t_1 = \{k : 0 \le k \le n_T - n - 1\}$ and $t_2 = \{k : n_T - n \le k \le n_T\}$, where n_T represents the counter for the current time at which the estimation is performed. Then the least-squares estimation problem for the entire time horizon (from $k = 0$ to $k = n_T$) can be approximated by the QP:

$$\hat{J}_{n_T} = \min_{\hat{\boldsymbol{x}}(n_T-n),\{\boldsymbol{\varepsilon}_1(k)\}} \left[(\hat{\boldsymbol{x}}(n_T-n) - \bar{\boldsymbol{x}}(n_T-n))^T \boldsymbol{P}^{-1}(n_T-n)(\hat{\boldsymbol{x}}(n_T-n) - \bar{\boldsymbol{x}}(n_T-n)) + \sum_{k=n_T-n}^{n_T-1} \boldsymbol{\varepsilon}_1^T(k)\boldsymbol{Q}^{-1}\boldsymbol{\varepsilon}_1(k) + \sum_{k=n_T-n+1}^{n_T} \boldsymbol{\varepsilon}_2^T(k)\boldsymbol{R}^{-1}\boldsymbol{\varepsilon}_2(k) + J^*_{n_T-n} \right] \tag{27}$$

with the equality constraints and update formulas:

$$\boldsymbol{\varepsilon}_1(k) = \hat{\boldsymbol{x}}(k+1) - \boldsymbol{A}\hat{\boldsymbol{x}}(k), \quad k = 0, \ldots, \quad n_T - 1 \tag{28}$$

$$\boldsymbol{\varepsilon}_2(k) = \boldsymbol{y}(k) - \hat{\boldsymbol{y}}(k), \quad k = 1, \ldots, \quad n_T \tag{29}$$

$$\bar{\boldsymbol{x}}(n_T-n) = \boldsymbol{A}\bar{\boldsymbol{x}}(n_T-n-1) + \boldsymbol{K}(n_T-n)(\boldsymbol{y}(n_T-n) - \boldsymbol{A}\bar{\boldsymbol{x}}(n_T-n-1)) \tag{30}$$

$$\boldsymbol{K}(n_T-n) = (\boldsymbol{A}\boldsymbol{P}(n_T-n-1)\boldsymbol{A}^T + \boldsymbol{Q})(\boldsymbol{R} + \boldsymbol{A}\boldsymbol{P}(n_T-n-1)\boldsymbol{A}^T + \boldsymbol{Q})^{-1} \tag{31}$$

$$\boldsymbol{P}(n_T-n) = (\boldsymbol{I} - \boldsymbol{K}(n_T-n))(\boldsymbol{A}\boldsymbol{P}(n_T-n-1)\boldsymbol{A}^T + \boldsymbol{Q}) \tag{32}$$

where the vector $\bar{\boldsymbol{x}}(n_T-n)$ is the Kalman filter estimate and the matrix $\boldsymbol{P}(n_T-n)$ is the corresponding covariance at time step n_T-n when initialized with estimate $\bar{\boldsymbol{x}}_0$ and covariance $\boldsymbol{P}_0$.

By using the decoupled state space system, the original QP can be converted to the element-wise QP,

$$\hat{J}_{n_T,i} = \min_{\hat{\bar{x}}_i(n_T-n),\{\tilde{\varepsilon}_{1,i}(k)\}} \left[\frac{(\hat{\bar{x}}_i(n_T-n) - \bar{\bar{x}}_i(n_T-n))^2}{\tilde{P}_{i,i}(n_T-n)} + \sum_{k=n_T-n}^{n_T-1} \frac{\tilde{\varepsilon}_{1,i}^2(k)}{\sigma_1} + \sum_{k=n_T-n+1}^{n_T} \frac{\tilde{\varepsilon}_{2,i}^2(k)}{\sigma_2} + J^*_{n_T-n,i} \right] \tag{33}$$

subject to

$$\tilde{\varepsilon}_{1,i}(k) = \hat{\bar{x}}_i(k+1) - \Sigma_{i,i}\hat{\bar{x}}_i(k) \tag{34}$$

$$\tilde{\varepsilon}_{2,i}(k) = \tilde{y}_i(k) - \hat{\tilde{y}}_i(k) \tag{35}$$

$$\bar{\bar{x}}_i(n_T-n) = \Sigma_{i,i}\bar{\bar{x}}_i(n_T-n-1) + \tilde{K}_{i,i}(n_T-n)(\tilde{y}_i(n_T-n) - \Sigma_{i,i}\bar{\bar{x}}_i(n_T-n-1)) \tag{36}$$

$$\tilde{K}_{i,i}(n_T-n) = \frac{\Sigma_{i,i}^2\tilde{P}_{i,i}(n_T-n-1) + \sigma_1}{\sigma_2 + \Sigma_{i,i}^2\tilde{P}_{i,i}(n_T-n-1) + \sigma_1} \tag{37}$$

$$\tilde{P}_{i,i}(n_T-n) = (1 - \tilde{K}_{i,i}(n_T-n))(\Sigma_{i,i}^2\tilde{P}_{i,i}(n_T-n-1) + \sigma_1) \tag{38}$$

where the $\bar{\bar{x}}_i(n_T-n)$ is the Kalman filter estimate and the $\tilde{P}_{i,i}(n_T-n)$ is the corresponding covariance at time step n_T-n when the Kalman filter for the ith decoupled system is initialized with estimate $\bar{\bar{x}}_{0,i}$ and covariance $\tilde{P}_{0,i,i}$ estimate at time step 0.

4. Decoupled constrained state estimation

Physically, all of the concentration values should be nonnegative. Including such constraints in the formulation would lead to a constrained discrete-time state estimation problem. A constrained state estimation problem is more difficult to handle than an unconstrained problem, which can be solved readily by Kalman filtering (Grewal & Andrews, 2001). The consideration of these constraints also differentiates this paper from the existing literature on state estimation of distributed parameter systems, which do not address state constraints. The MHE method is suitable for this purpose as it reformulates the state estimation problem as QP, which naturally allows for inclusion of inequality constraints (Rao et al., 2001; Robertson et al., 1996).

4.1. Constrained moving horizon estimation

For the state-space system of Eqs. (12) and (13), the constrained linear state estimation problem can be formulated as a quadratic problem with inequality constraints,

$$\mathbf{0} \le \hat{\boldsymbol{x}}(k) \le \boldsymbol{x}_{\max}, \quad k = 0, \ldots, n_T \tag{39}$$

The reason for the positivity constraint for the concentration value is obvious. On the other hand, the upper bound $\boldsymbol{x}_{\max}$ is provided in order to make the feasible region compact (a property needed for the later ellipsoidal approximation). Such an upper bound can be derived from first-principles or practical considerations in most state estimation problems for distributed parameter systems.

The relationship between the estimates $\{\hat{\boldsymbol{x}}(k)\}$ and the solution pair $(\boldsymbol{x}_0, \{\boldsymbol{\varepsilon}_1(k)\})$ in the QP can be derived from Eq. (28) to give

$$\begin{bmatrix} \hat{\boldsymbol{x}}(n_T-n) \\ \hat{\boldsymbol{x}}(n_T-n+1) \\ \hat{\boldsymbol{x}}(n_T-n+2) \\ \vdots \\ \hat{\boldsymbol{x}}(n_T) \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & 0 & 0 & \cdots & 0 \\ \boldsymbol{A} & \boldsymbol{I} & 0 & \ddots & \vdots \\ \boldsymbol{A}^2 & \boldsymbol{A} & \boldsymbol{I} & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \boldsymbol{A}^n & \boldsymbol{A}^{n-1} & \boldsymbol{A}^{n-2} & \cdots & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{x}}(n_T-n) \\ \boldsymbol{\varepsilon}_1(n_T-n) \\ \boldsymbol{\varepsilon}_1(n_T-n+1) \\ \vdots \\ \boldsymbol{\varepsilon}_1(n_T-1) \end{bmatrix} \tag{40}$$

By inserting Eq. (40) into Eqs. (28), (29) and (39) and Eqs. (28) and (29) into the objective function (27), the final QP takes a reduced form with only inequality constraints:

$$\min_{\boldsymbol{z}} \boldsymbol{z}(n_T)^T \boldsymbol{H}(n_T)\boldsymbol{z}(n_T) + \boldsymbol{f}^T(n_T)\boldsymbol{z}(n_T) \tag{41}$$

subject to

$$\mathbf{0} \le \boldsymbol{S}(n_T)\boldsymbol{z}(n_T) \le \boldsymbol{x}_{\max} \tag{42}$$

where $\boldsymbol{z}, \boldsymbol{H}, \boldsymbol{f}$, and $\boldsymbol{S}$ are defined by

$$\boldsymbol{z}(n_T) = \begin{bmatrix} \hat{\boldsymbol{x}}(n_T-n) \\ \boldsymbol{\varepsilon}_1(n_T-n) \\ \boldsymbol{\varepsilon}_1(n_T-n+1) \\ \vdots \\ \boldsymbol{\varepsilon}_1(n_T-1) \end{bmatrix} \tag{43}$$

$$\begin{aligned} \boldsymbol{H}(n_T) = & \begin{bmatrix} \boldsymbol{P}(n_T-n)^{-1} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}^{-1} & \mathbf{0} & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \boldsymbol{Q}^{-1} & \ddots & \mathbf{0} \\ \vdots & \ddots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \cdots & \boldsymbol{Q}^{-1} \end{bmatrix} + \begin{bmatrix} \boldsymbol{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \boldsymbol{A} & \boldsymbol{I} & \mathbf{0} & \ddots & \vdots \\ \boldsymbol{A}^2 & \boldsymbol{A} & \boldsymbol{I} & \ddots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \mathbf{0} \\ \boldsymbol{A}^n & \boldsymbol{A}^{n-1} & \boldsymbol{A}^{n-2} & \cdots & \boldsymbol{I} \end{bmatrix}^T \\ & \times \begin{bmatrix} \boldsymbol{R}^{-1} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{R}^{-1} & \mathbf{0} & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \boldsymbol{R}^{-1} & \ddots & \mathbf{0} \\ \vdots & \ddots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \cdots & \boldsymbol{R}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \boldsymbol{A} & \boldsymbol{I} & \mathbf{0} & \ddots & \vdots \\ \boldsymbol{A}^2 & \boldsymbol{A} & \boldsymbol{I} & \ddots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \mathbf{0} \\ \boldsymbol{A}^n & \boldsymbol{A}^{n-1} & \boldsymbol{A}^{n-2} & \cdots & \boldsymbol{I} \end{bmatrix} \end{aligned} \tag{44}$$

$$\boldsymbol{f}(n_T) = -\begin{bmatrix} \tilde{\boldsymbol{x}}(n_T-n)\boldsymbol{P}(n_T-n)^{-1} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mathbf{0} \\ \boldsymbol{y}(n_T-n+1)^T\boldsymbol{R}^{-1} \\ \boldsymbol{y}(n_T-n+2)^T\boldsymbol{R}^{-1} \\ \vdots \\ \boldsymbol{y}(n_T)^T\boldsymbol{R}^{-1} \end{bmatrix} \tag{45}$$

$$\boldsymbol{S}(n_T) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{A} & \boldsymbol{I} & \boldsymbol{0} & \ddots & \vdots \\ \boldsymbol{A}^2 & \boldsymbol{A} & \boldsymbol{I} & \ddots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \boldsymbol{0} \\ \boldsymbol{A}^n & \boldsymbol{A}^{n-1} & \boldsymbol{A}^{n-2} & \cdots & \boldsymbol{I} \end{bmatrix} \tag{46}$$

4.2. Ellipsoid constraint approximation

For the ith decoupled state-space system, the reduced QP is

$$\min_{\tilde{\boldsymbol{z}}_i} \tilde{\boldsymbol{z}}_i(n_T)^T \tilde{\boldsymbol{H}}_i(n_T)\tilde{\boldsymbol{z}}_i(n_T) + \tilde{\boldsymbol{f}}_i(n_T)^T \tilde{\boldsymbol{z}}_i(n_T), \tilde{\boldsymbol{z}}(n_T) = \boldsymbol{U}^T \boldsymbol{z}(n_T) \tag{47}$$

where $\tilde{\boldsymbol{H}}_i$ and $\tilde{\boldsymbol{f}}_i$ are defined by

$$\begin{aligned} \tilde{\boldsymbol{H}}_i(n_T) = & \begin{bmatrix} \tilde{P}_{i,i}(n_T-n)^{-1} & 0 & 0 & \cdots & 0 \\ 0 & \sigma_1^{-1} & 0 & \ddots & \vdots \\ 0 & 0 & \sigma_1^{-1} & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & \sigma_1^{-1} \end{bmatrix} + \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ \Sigma_{i,i} & I & 0 & \ddots & \vdots \\ \Sigma_{i,i}^2 & \Sigma_{i,i} & I & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \Sigma_{i,i}^n & \Sigma_{i,i}^{n-1} & \Sigma_{i,i}^{n-2} & \cdots & I \end{bmatrix}^T \\ & \times \begin{bmatrix} \sigma_2^{-1} & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2^{-1} & 0 & \ddots & 0 \\ 0 & 0 & \sigma_2^{-1} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & \sigma_2^{-1} \end{bmatrix} \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ \Sigma_{i,i} & I & 0 & \ddots & \vdots \\ \Sigma_{i,i}^2 & \Sigma_{i,i} & I & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \Sigma_{i,i}^n & \Sigma_{i,i}^{n-1} & \Sigma_{i,i}^{n-2} & \cdots & I \end{bmatrix} \end{aligned} \tag{48}$$

$$\boldsymbol{f}_i(n_T) = \begin{bmatrix} \tilde{\tilde{x}}_i(n_T-n)\tilde{P}_{i,i}(n_T-n)^{-1} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ \tilde{y}_{1,i}(n_T-n+1)\sigma_2^{-1} \\ \tilde{y}_{2,i}(n_T-n+2)\sigma_2^{-1} \\ \vdots \\ \tilde{y}_{n,i}(n_T)\sigma_2^{-1} \end{bmatrix} \tag{49}$$

For using the original constraint in the QP based on the decoupled state-space system, the constraint can be reformulated with respect to the coordinate-transformed state variable, $\tilde{\boldsymbol{z}}$:

$$\boldsymbol{0} \le \tilde{\boldsymbol{S}}(n_T)\tilde{\boldsymbol{z}}(n_T) \le \boldsymbol{x}_{\max}, \tag{50}$$

$$\tilde{\boldsymbol{S}}(n_T) = \begin{bmatrix} \boldsymbol{U} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{U} & \boldsymbol{0} & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U} & \ddots & \boldsymbol{0} \\ \vdots & \ddots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \cdots & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{\Sigma} & \boldsymbol{I} & \boldsymbol{0} & \ddots & \vdots \\ \boldsymbol{\Sigma}^2 & \boldsymbol{\Sigma} & \boldsymbol{I} & \ddots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \ddots & \boldsymbol{0} \\ \boldsymbol{\Sigma}^n & \boldsymbol{\Sigma}^{n-1} & \boldsymbol{\Sigma}^{n-2} & \cdots & \boldsymbol{I} \end{bmatrix} \tag{51}$$

Unfortunately, it is not possible to apply the transformed inequality linear constraint set (50) to the decoupled system directly since the inequality constraint involves a non-symmetric matrix $\tilde{\boldsymbol{S}}(n_T)$ in Eq. (51) and cannot be separated into a decoupled form using the SVD. This problem can be resolved by approximating the linear constraint by an *ellipsoidal constraint* (VanAntwerp & Braatz, 2000b). The basic idea is to compute the maximal volume ellipsoid completely contained within the polytope defined by

$$\boldsymbol{F}\tilde{\boldsymbol{z}}(k) \le \boldsymbol{q} \tag{52}$$

$$\boldsymbol{F} = \begin{bmatrix} -\tilde{\boldsymbol{S}} \\ \tilde{\boldsymbol{S}} \end{bmatrix} \tag{53}$$

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{x}_{\max} \end{bmatrix} \tag{54}$$

The region defined by the ellipsoidal constraint is exactly inscribed in the original polytope constraint set and therefore represents a feasible solution region. The ellipsoid E (in the transformed coordinates) can be expressed as

$$E = \{\tilde{\boldsymbol{z}} | \tilde{\boldsymbol{z}} = \boldsymbol{B}\boldsymbol{y} + \boldsymbol{d}, \boldsymbol{y}^T\boldsymbol{y} \leq 1\} \tag{55}$$

where $\boldsymbol{d}$ is a vector that defines the center of the ellipsoid and $\boldsymbol{B}$ is a diagonal matrix that defines the relative lengths of the axes of the ellipsoid. The restriction of a diagonal matrix can be relaxed to allow $\boldsymbol{B}$ to be a full matrix, but is made here to simplify the step of applying the approximate penalty method to derive an analytical solution later. Finding the largest ellipsoid that is completely contained in the polytope region is formulated as the convex optimization (Bland, Goldfarb, & Todd, 1981):

$$\min_{\boldsymbol{B},\boldsymbol{d}} \{-\log\det(\boldsymbol{B})\} \tag{56}$$

subject to the constraints

$$\boldsymbol{B} > 0 \tag{57}$$

$$||\boldsymbol{B}\boldsymbol{F}_i(k)|| + \boldsymbol{F}_i^T(k)\boldsymbol{d} \leq q_i, \quad \forall i \in [1, 2, \ldots, n_X n_Y] \tag{58}$$

that is solved offline, where $\boldsymbol{F}_i^T(k)$ is a row vector corresponding to the ith row of $\boldsymbol{F}(k)$ and q_i is the ith element of $\boldsymbol{q}$. This optimization can be solved using the SeDuMi with the Matlab interface *cvx* (Grant, Boyd, & Ye, 2006). The ellipsoidal constraint form can be converted to

$$(\tilde{\boldsymbol{z}} - \boldsymbol{d})^T\boldsymbol{B}^{-2}(\tilde{\boldsymbol{z}} - \boldsymbol{d}) \leq 1 \tag{60}$$

which can be expressed as the sum of each element for $\tilde{\boldsymbol{z}}_i$,

$$\sum_{i=1}^{n_X n_Y} (\tilde{\boldsymbol{z}}_i - \boldsymbol{d}_i)^T\boldsymbol{B}_i^{-2}(\tilde{\boldsymbol{z}}_i - \boldsymbol{d}_i) \leq 1 \tag{61}$$

where $\boldsymbol{B}_i$ is the diagonal matrix that is the ith diagonal block of $\boldsymbol{B}$.

The step of finding the optimal ellipsoidal constraint in Eq. (56) requires significant computational and random access memory (RAM) requirements for a high dimensional system. In the case of a 100×100 system, the optimization problem (56) contains more than 30,000 variables and almost 30,000 constraints. The RAM requirement can be greatly reduced by using sparse linear algebra.

4.3. Penalty method

The reformulated QP with the approximated ellipsoid constraint can be efficiently solved by the penalty method, which adds the constraint (61) to the objective function (47) as a penalty term to assign a high cost to infeasible points:

$$\min_{\tilde{z}_i, i=1,\ldots,n_X n_Y} \sum_{i=1}^{n_X n_Y} \left\{ \frac{1}{2}\tilde{\boldsymbol{z}}_i^T\tilde{\boldsymbol{H}}_i\tilde{\boldsymbol{z}}_i + \tilde{\boldsymbol{f}}_i^T\tilde{\boldsymbol{z}}_i + \lambda \left\{ (\tilde{\boldsymbol{z}}_i - \boldsymbol{d}_i)^T\boldsymbol{B}_i^{-2}(\tilde{\boldsymbol{z}}_i - \boldsymbol{d}_i) - \frac{1}{n_X n_Y} \right\} \right\} \tag{62}$$

First this optimization is solved for $\lambda = 0$. Then the solution to the unconstrained optimization problem can be calculated by

$$\tilde{\boldsymbol{z}}_i = -\tilde{\boldsymbol{f}}_i^T\tilde{\boldsymbol{H}}_i^{-1} \tag{63}$$

which can be checked against the original linear constraints. If the unconstrained solution satisfies these constraints, then it is implemented as the solution. On the other hand, if the unconstrained solution violates the constraints, then $\lambda > 0$ and a global solution can be obtained as

$$\tilde{\boldsymbol{z}}_i = (-\tilde{\boldsymbol{f}}_i^T + \lambda\boldsymbol{d}_i^T\boldsymbol{B}_i^{-2})(\tilde{\boldsymbol{H}}_i + \lambda\boldsymbol{B}_i^{-2})^{-1} \tag{64}$$

In standard ellipsoidal algorithms, a new ellipsoid that encloses the optimal solution is recomputed and added to the objective function at each time step. These updates oftentimes offer little performance gain while significantly increasing the computational cost, however.

In our QP formulation with time-invariant inequality constraints, the ellipsoidal constraint can be computed once off-line, which makes the ellipsoidal constraint approximation approach reasonable for the fast MHE method. The only (potential) iteration step performed on-line is to adjust λ by a ratio until the original constraints are satisfied.

This idea is related to the approximate primal penalty method, which has been used in fast MPC (VanAntwerp & Braatz, 2000a; Wang & Boyd, 2010). MPC uses only one manipulated variable value at time step n_T although the solution set contains the manipulated variable values until time step $n_T + m$, where m is a control prediction horizon. In a similar manner, MHE uses only the state estimate value at time step n_T from the solution set containing the estimate values from time $n_T - n$.

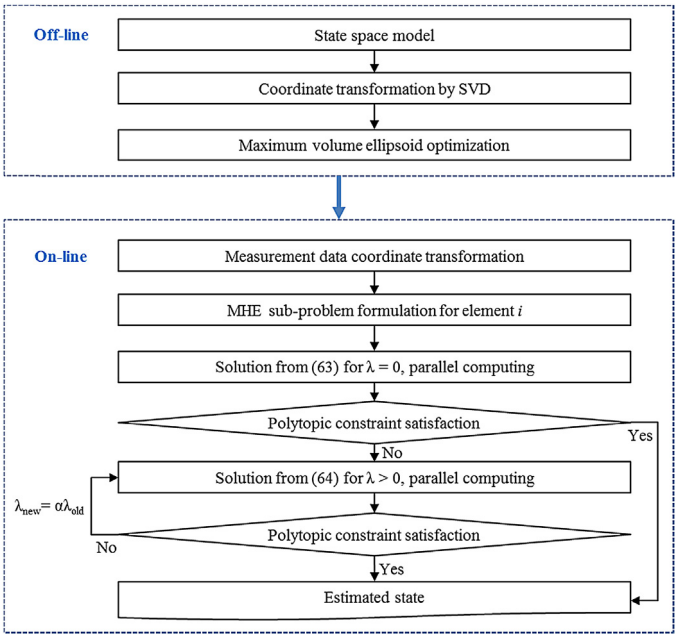

Fig. 3. Flowchart of a parallel computing algorithm for moving-horizon state estimation.

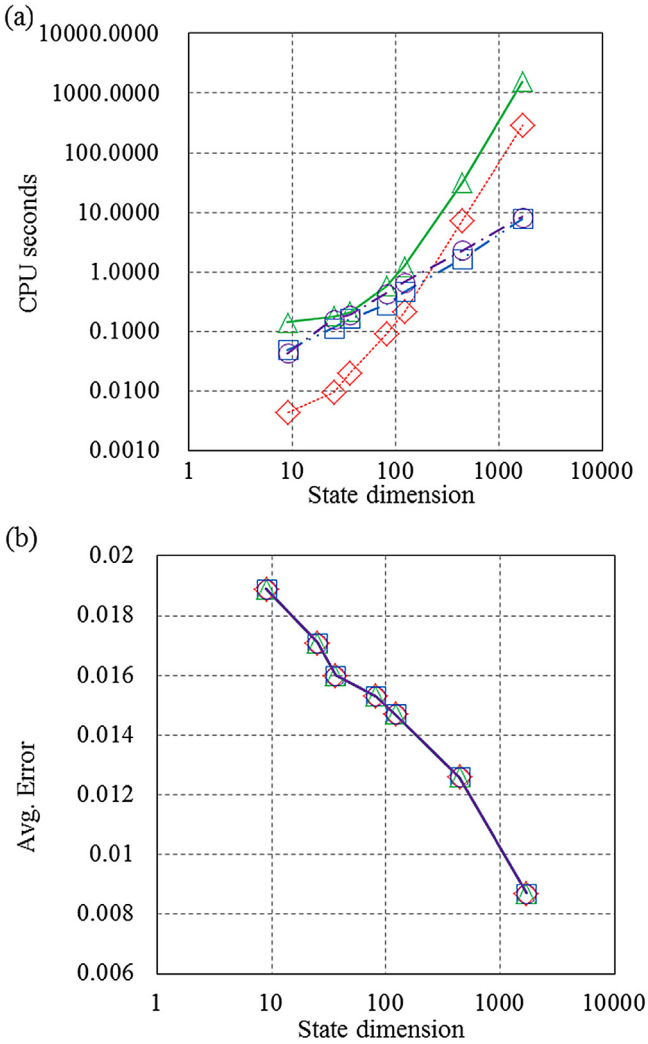

Fig. 4. The computation time (a) and average error (b) among standard KF (diamond, ●●●), decoupled KF (square, –●–), standard MHE (triangle, —), and decoupled MHE (circle, –●–) with increasing state dimension for the unconstrained state estimation problem. Lines used only to guide the eye.

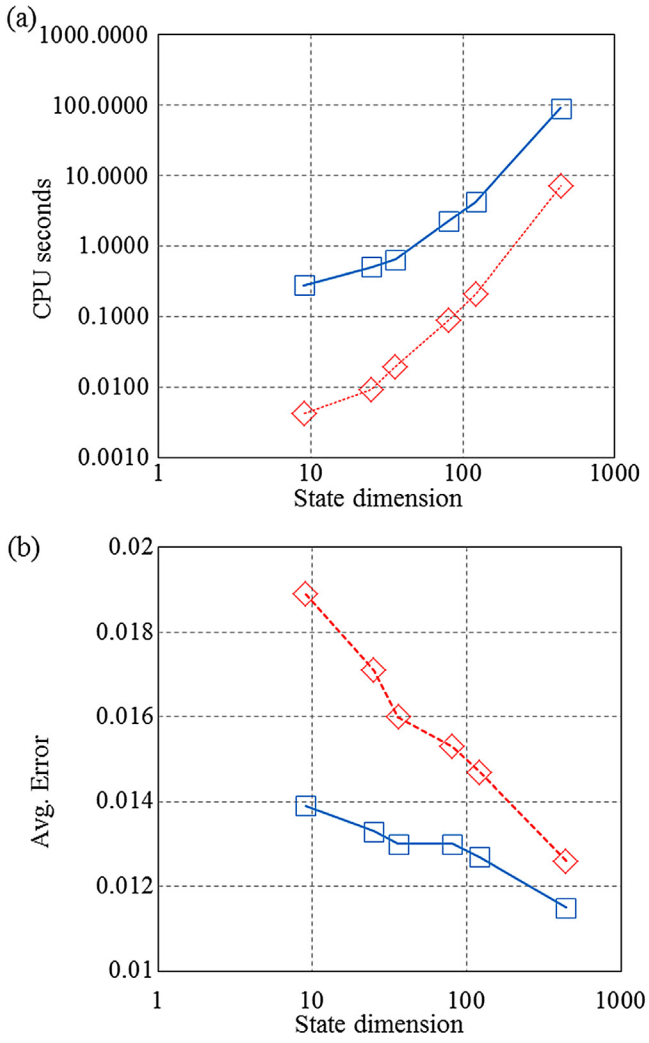

Fig. 5. The computation time (a) and average error (b) for the unconstrained MHE (diamond, ●●●) and constrained MHE (square, —) with increasing state dimension. Lines used only to guide the eye.

4.4. Parallel computing algorithm

Our reformulation of MHE speeds up the computation and lowers the memory requirement significantly by converting the original problem having dimensions $n_X n_Y n \times n_X n_Y n$ into $n_X n_Y$ smaller sub-problems of $n \times n$ dimensions. The sub-problems are perfectly decoupled, which allow them to be computed in parallel. A flowchart of the overall algorithm is in Fig. 3. The algorithm is divided into two parts: off-line and on-line. In the off-line part, the state-space model for the sensor array system is defined and the system coordinate is transformed via SVD to decouple the system. Then the maximum volume ellipsoid contained within the original polytopic constraint is obtained by solving the convex optimization. In the on-line part, first, measurement data from the sensors are gathered and transformed according to the model coordinate. The MHE sub-problems are formulated for each element i of the transformed state variable. The solution is obtained by Eq. (63) for $\lambda = 0$ and checked against the original polytopic constraints. If the constraints are not satisfied, the solution is refined by solving Eq. (64) for $\lambda > 0$ and λ is decreased with a factor $\alpha < 1$ until the constraints are satisfied.

Although the ellipsoidal-constraint approximation makes the parallel implementation of the on-line MHE calculation possible by converting the original problem into independent smaller sub-problems, the approximation also leads to some sacrifice in the accuracy. Furthermore, the off-line calculation for generating the ellipsoidal constraint becomes exponentially heavier with increasing state dimension and can become a bottleneck.

5. Simulation and discussion

The performance of the proposed fast MHE method is compared with that of the conventional KF and the MHE without or with linear constraints. For this purpose, a simple but realistic system is chosen, which is representative of a carbon nanotube-based sensor array system.

5.1. CNT-based sensor array system and simulation study

When target molecules adsorb on the surface of an individual CNT sensor, the fluorescence is partially quenched resulting in a discretized decrease in light intensity. Similarly, the light intensity increases when molecules desorb from the surface. By counting the number of step

Table 1
Comparison of the computation time and average error values between the original MHE and the proposed MHE with increasing horizon length.

$n_x n_y$	Original QP-based MHE		Fast MHE (quad core)	
	Time	Average error	Time	Average error
9	0.2799	0.0139	0.0287	0.0209
25	0.5044	0.0133	0.0761	0.0162
36	0.6445	0.0130	0.1101	0.0151
81	2.2811	0.0130	0.2559	0.0138
121	4.2050	0.0127	0.4003	0.0132

changes in the intensity, the number of adsorbed molecules and thus the bulk concentration surrounding the sensor can be deduced (Cognet et al., 2007; Zhang et al., 2010).

Synthesized CNT-based sensors can be placed in a two-dimensional sensor array covering an area of up to ten or more squared centimeters (Zhang et al., 2010). Such a sensor array can be designed so that the sensors are distributed throughout a two-dimensional field (Fig. 1). From this array system, the objective is to estimate time profiles of two-dimensional concentration fields of target molecules. This spatiotemporal profile information can be useful for monitoring and control of a 2D spatial system (e.g., see Kishida & Braatz, 2009, and citations therein).

In the simulation study that follows, the measurement values are assumed to be obtained every 0.1 time step from an array of sensors, which are distributed uniformly with possible distances of 0.5, 0.25, 0.2, 0.125, 0.1 along the x and y directions, the ranges of which are normalized to be 0 to 1. Note that, since the spacing in the sensor array system dictates the discretization resolution, the dimensions of the system variables and matrices become higher as the distances between the sensors become smaller.

For the MHE methods, the horizon length is fixed with 1 for fast estimation and also few or no improvement in performance of the state estimation was observed when the horizon length is more than 1 for this linear system as the particular way of updating the initial

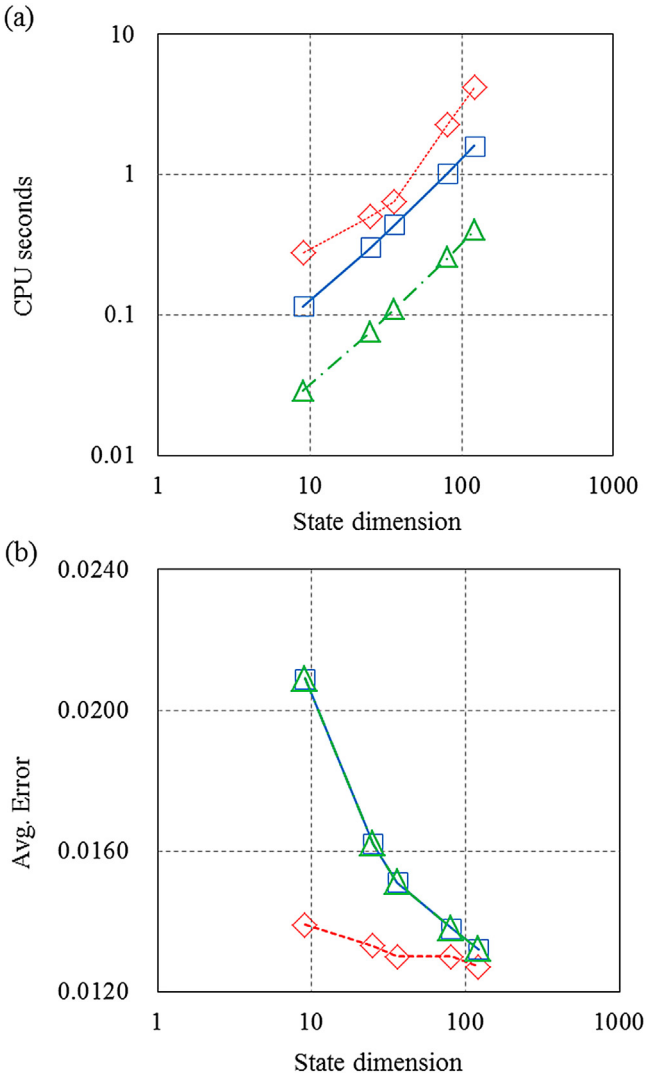

Fig. 6. The CPU time (top) and average error (bottom) for original MHE (diamond, ●●●), proposed fast MHE (square, —), and proposed fast MHE solved in parallel on the Quad core CPU (triangle, –●–) with increasing state dimension. Lines used only to guide the eye.

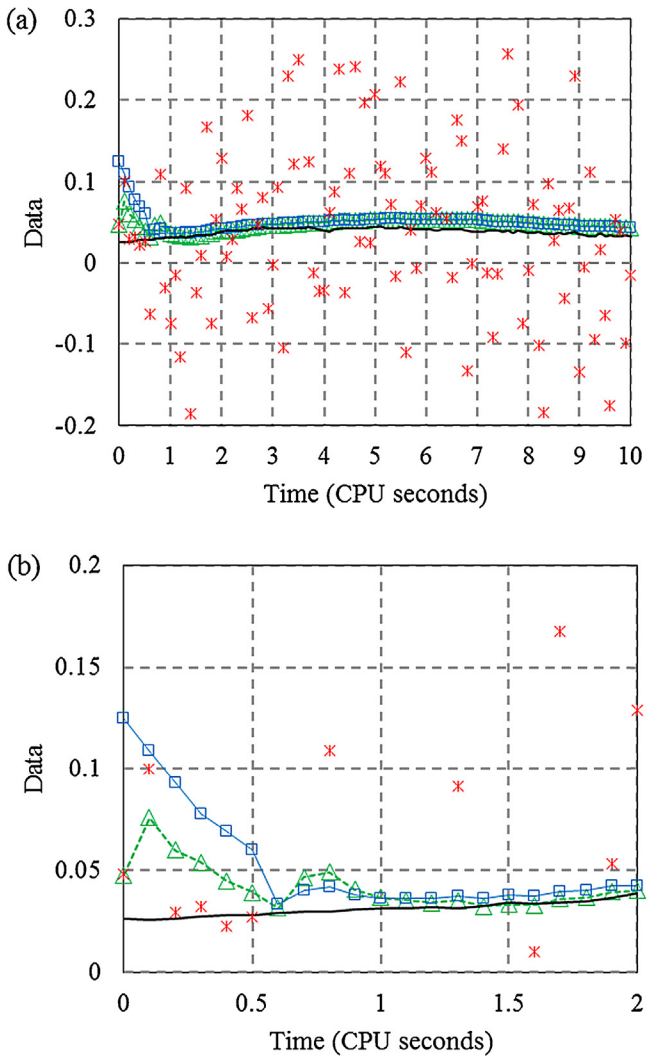

Fig. 7. Plots of the estimated vs. true state and noise data for constrained discrete-time state estimation using the MHE QP (triangle) and the proposed MHE QP (square) for $\Delta x = \Delta y = 0.2$ at the position $(x, y) = (0.6, 0.6)$ on the two-dimensional field (true state: black solid line; noisy data: red star marker) for (a) the whole time period and (b) a zoom-in of early time. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

error weighting matrix makes the unconstrained MHE equivalent to the Kalman filter regardless of the horizon size (Rao et al., 2001). The covariance parameters, $\boldsymbol{Q}$, $\boldsymbol{R}$, and $\boldsymbol{P}_0$, appearing in the objective function are chosen as 0.000001 $\boldsymbol{I}$, 0.01 $\boldsymbol{I}$, and $\boldsymbol{I}$, respectively. The constants in front of the identity matrices in $\boldsymbol{Q}$ and $\boldsymbol{R}$ correspond to the variances of the process and observation noises. In the particular case tested, the observation noise is assumed to be much larger than the process noise. The matrix $\boldsymbol{P}_0$ is chosen sufficiently large to reflect the uncertainty in the initial estimate. After time step $k \geq n$, the $\boldsymbol{P}$ is updated by the Kalman filter covariance update formula.

5.2. Unconstrained state estimation

As explained, the two-dimensional diffusion equation is discretized and then decomposed into multiple independent single dimension systems via SVD. In this study, the performance of the decoupled KF and MHE are compared with the standard KF and MHE.

Fig. 4a compares the computational costs for the four state estimation methods for systems of increasing state dimension. The computational costs for the standard KF and standard MHE increased as a quadratic function of the system dimension, whereas the costs for the decoupled KF and MHE methods increased linearly. The computational speed could be further enhanced by employing parallel computers.

Fig. 4b compares the average errors represented by the mean absolute error (from now on, average error means mean absolute error) for the four state estimation methods for systems of increasing state dimension. The average errors decreased with increasing state dimension, as higher dimensions have a higher number of sensors. The average error values are almost same for the four state estimation methods. The coordinate transformation has no effect on the accuracy of the methods. The result is consistent with the fact that the unconstrained MHE for a linear state-space system corresponds to the linear KF (Muske et al., 1993; Rao et al., 2001).

5.3. Constrained state estimation

Ignoring constraints can produce state estimates that are physically unrealistic (e.g., negative concentration values) due to observation noises. In Fig. 5a, it can be observed that the computational costs of constrained MHE are significantly higher when compared to

unconstrained MHE. However, in Fig. 5b, it can be observed that the average error of constrained MHE is significantly less than unconstrained MHE due to the imposition of the nonnegativity constraints.

The computational savings and possible performance loss from the use of fast MHE (with the ellipsoidal constraint approximation) are examined next. For solving the standard QP-based MHE, several representative solvers in MATLAB capable of handling sparse matrices were examined. The 'interior-point-convex' algorithm in *quadprog* (Matlab Optimization Toolbox) was found to be the fastest solver for the studied system having high-dimensional matrices. In the case of fast MHE, instead of using a general QP solver in MATLAB, its solution was obtained using the parallel algorithm in Section 4.4. The optimizations were run on a Quad core 3.10 GHz CPU workstation with 8 GB RAM.

The initial state estimate $\boldsymbol{x}_0$ was set as the zero vector in both approaches. Different state dimensions were tried, ranging from $n_X n_Y = 25$ for $\Delta x = \Delta y = 0.25$ to $n_X n_Y = 121$ for $\Delta x = \Delta y = 0.1$. The penalty parameter was decreased by a ratio $\alpha < 1$ (for example, 0.95) each time until the constraint was satisfied. The computation time and the average error between the true states and the state estimates are summarized in Table 1 and plotted in Fig. 6. Plots of the true state vs. the estimates from the two approaches are shown in Fig. 7.

First of all, the results in Fig. 6a and Table 1 show that much lower computation time was needed for the proposed fast MHE approach. In addition, the computation time for the proposed MHE grew much more slowly as a function of the state dimension than for the original QP-based MHE. Fig. 6a shows problem size vs. computation time for the two approaches. The original QP-based MHE follows a quadratic slope, whereas the proposed fast MHE follows a linear slope. The reason for the decrease in the computation time is that the size of the QP to be solved at each time step is reduced by a factor of $n_X n_Y \times n_X n_Y$. The reduction in the computation time is not exactly by $n_X n_Y \times n_X n_Y$ because the algorithms to find the optimal solution are different in the two methods. Nevertheless, the computational savings are significant and the gap widens fast when the state dimension is increased.

Fig. 6b and Table 1 compare the average error values for the two different MHE methods. Both methods show decreasing average errors with increasing state dimension. The reason is that more state variable and measurement data improve the performance of the state estimation. However, the proposed MHE method shows a slightly poorer performance than the conventional MHE method (Fig. 6b). The proposed MHE method employs a suboptimal solution due to the approximation of the polytopic constraint with an ellipsoidal constraint, which is more restrictive. The errors become negligible as the number of sensors increase in the sensor array.

The difference in the estimation performance between the proposed fast MHE method and the conventional MHE method can be examined more closely in Fig. 7, which plots the estimated responses from the initial time step to the final time step. The plot of the state estimates shows that the proposed MHE method gives slightly poorer estimation results at the early stage, but then converges to being essentially the same as for the conventional MHE method.

6. Conclusions

By adopting an idea from the "fast MPC" in the context of state estimation, a fast MHE method is developed that dramatically reduces the computational time of the online optimization. The proposed method uses the singular value decomposition to transform a very large QP with linear constraints into a large set of decoupled QPs with a single ellipsoidal constraint. To enforce the ellipsoidal constraint in the reformulated QP, a penalty method with a decreasing penalty parameter can be used to obtain an analytical solution. This method was applied to a two-dimensional diffusion system and the results showed significant improvement in the computational speed with little performance loss when compared to the original MHE QP.

The current limiting factor in applying the proposed method to a large-scale system may be the calculation of the maximal volume ellipsoid inscribed within a given polytope, even though this is a convex optimization to be solved offline. An efficient way to obtain such as an approximate ellipsoidal constraint is needed. Another issue is that the current manufacturing technology for the carbon nanotube-based sensor arrays that motivated this project distributes these sensors randomly on a tiny two-dimensional field (Zhang et al., 2010). If their placements cannot be controlled perfectly, this results in some missing information for a uniform mesh grid. Hence the assumption that concentrations are measured at all nodes of the discretized mesh is limiting and needs to be relaxed, or two-dimensional nanopositioners need to be developed that precisely place the nanosensors on a uniform grid (Chu & Gianchandani, 2003; Sebastian & Salapaka, 2005).

Acknowledgments

This work was supported by the Advanced Biomass R&D Center (ABC) of Global Frontier Project funded by the Ministry of Education, Science and Technology (ABC-0031354), the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (20110006839).

References

Bland, R. G., Goldfarb, D., & Todd, M. J. (1981). The ellipsoid method: A survey. *Operations Research*, *29*, 1039–1091.
Boghossian, A. A., Zhang, J., Barone, P. W., Reuel, N. F., Kim, J.-H., Heller, D. A., et al. (2011). Near-infrared fluorescent sensors based on single-walled carbon nanotubes for life sciences applications. *ChemSusChem*, *4*, 848–863.
Chu, L. L., & Gianchandani, Y. B. (2003). A micromachined 2D positioner with electrothermal actuation and sub-nanometer capacitive sensing. *Journal of Micromechanics and Microengineering*, *13*, 279–285.
Curtain, R. F., & Zwart, H. (1995). *An introduction to infinite-dimensional linear systems theory*. New York: Springer.
Cognet, L., Tsyboulski, D. A., Rocha, J. D. R., Doyle, C. D., Tour, J. M., & Weisman, R. B. (2007). Stepwise quenching of exciton fluorescence in carbon nanotubes by single-molecule reactions. *Science*, *316*, 1465–1468.
Grant, M., Boyd, S., & Ye, Y. (2006). *CVX: Matlab software for disciplined convex programming*. http://www.stanford.edu/~boyd/cvx
Grewal, M. S., & Andrews, A. P. (2001). *Kalman filtering: Theory practice using MATLAB* (2nd ed.). New York: Wiley.
Haberman, R. (2004). *Applied partial differential equations with Fourier series and boundary value problems* (4th ed.). New Jersey: Pearson.
Jin, H., Heller, D. A., Kalbacova, M., Kim, J.-H., Zhang, J., Boghossian, A. A., et al. (2010). Detection of single-molecule H_2O_2 signalling from epidermal growth factor receptor using fluorescent single-walled carbon nanotubes. *Nature Nanotechnology*, *5*, 302–309.
Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, *82*, 35–45.
Kalman, R. E., & Bucy, R. S. (1961). New results in linear filtering and prediction theory. *Transactions of the ASME, Journal of Basic Engineering*, *93*, 95–108.
Kishida, M., & Braatz, R. D. (2009). Optimal spatial field control of distributed parameter systems. In *Proceedings of the American control conference* St Louis, MO, (pp. 32–37).

Kishida, M., Pack, D. W., & Braatz, R. D. (2010). State-constrained optimal spatial field control for controlled release in tissue engineering. In *Proceedings of the American control conference* Baltimore, MD, (pp. 4361–4366).
Morton, K. W., & Mayers, D. F. (1995). *Numerical solution of partial differential equations*. Cambridge: Cambridge University Press.
Muske, K. R., Rawlings, J. B., & Lee, J. H. (1993). Receding horizon recursive state estimation. In *Proceedings of the American control conference* San Francisco, (pp. 900–904).
Rao, C. V., Rawlings, J. B., & Lee, J. H. (2001). Constrained linear state estimation—A moving horizon approach. *Automatica*, *37*, 1619–1628.
Robertson, D. G., Lee, J. H., & Rawlings, J. B. (1996). A moving horizon-based approach for least-squares state estimation. *AIChE Journal*, *42*, 2209–2224.
Russell, D. L. (1967). Nonharmonic Fourier series in the control theory of distributed parameter systems. *Journal of Mathematical Analysis and Applications*, *18*, 542–560.
Sebastian, A., & Salapaka, S. M. (2005). Design methodologies for robust nano-positioning. *IEEE Transactions on Control Systems and Technology*, *13*, 868–876.
VanAntwerp, J. G., & Braatz, R. D. (2000a). Fast model predictive control of sheet and film processes. *IEEE Transactions on Control Systems and Technology*, *8*, 408–417.
VanAntwerp, J. G., & Braatz, R. D. (2000b). Model predictive control of large scale processes. *Journal of Process Control*, *10*, 1–8.
Wang, Y., & Boyd, S. (2010). Fast model predictive control using online optimization. *IEEE Transactions on Control Systems and Technology*, *18*, 267–278.
Zhang, J., Boghossian, A. A., Barone, P. W., Rwei, A., Kim, J.-H., Lin, D., et al. (2010). Single molecule detection of nitric oxide enabled by $d(AT)_{15}$ DNA adsorbed to near infrared fluorescent single-walled carbon nanotubes. *Journal of the American Chemical Society*, *133*, 567–581.