Applied Mathematical Modelling 51 (2017) 605–625

Contents lists available at ScienceDirect

Applied Mathematical Modelling

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

Identification of distributed-parameter systems from sparse measurements

Z. Hidayat[a,*], R. Babuška[a], A. Núñez[b], B. De Schutter[a]

[a] Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, The Netherlands
[b] Section of Railway Engineering, Delft University of Technology, Stevinweg 1, 2628 CN Delft, The Netherlands

ARTICLE INFO

Article history:
Received 2 July 2015
Revised 28 April 2017
Accepted 3 July 2017
Available online 6 July 2017

Keywords:
System identification
Finite-difference method
Input selection
Indoor climate modeling
Greenhouse climate model

ABSTRACT

In this paper, a method for the identification of distributed-parameter systems is proposed, based on finite-difference discretization on a grid in space and time. The method is suitable for the case when the partial differential equation describing the system is not known. The sensor locations are given and fixed, but not all grid points contain sensors. Per grid point, a model is constructed by means of lumped-parameter system identification, using measurements at neighboring grid points as inputs. As the resulting model might become overly complex due to the involvement of neighboring measurements along with their time lags, the Lasso method is used to select the most relevant measurements and so to simplify the model. Two examples are reported to illustrate the effectiveness of the method, a simulated two-dimensional heat conduction process and the construction of a greenhouse climate model from real measurements.

1. Introduction

Many real-life processes are distributed-parameter systems. Examples include chip manufacturing plants [1]; process control systems such as packed-bed reactors [2], reverse flow reactors [3], and waste water treatment plants [4]; flexible structures in atomic force microscopes [5]; ultraviolet disinfection installations in the food industry [6]; electrochemical processes [7]; or drying installations [8].

Distributed-parameter systems are typically modeled using partial differential equations. However, developing such models from first principles is a tedious and time-consuming process [9]. If input–output measurements are available, a model can be constructed by using system identification methods. However, due to the large number of spatially interdependent state variables, the identification of distributed-parameter systems is considerably more complex than the identification of lumped-parameter systems, and it is known to be an ill-posed inverse problem [10] because the solution is not unique [11]. There are three main approaches to the identification of distributed-parameter systems [12]: (i) direct identification, (ii) reduction to a lumped-parameter system, and (iii) reduction to an algebraic equation. The direct identification approach uses the infinite-dimensional system model to find the parameters of the systems. This case includes identification of a certain parameter of interest related to an application, e.g., heat conduction [13–15]. The reduction-based approaches involve spatial

[*] Corresponding author.
E-mail addresses: z.hidayat@tudelft.nl, zulkifli@ee.its.ac.id (Z. Hidayat), r.babuska@tudelft.nl (R. Babuška), a.a.nunezvicencio@tudelft.nl (A. Núñez), b.deschutter@tudelft.nl (B. De Schutter).

http://dx.doi.org/10.1016/j.apm.2017.07.001
0307-904X/© 2017 Elsevier Inc. All rights reserved.

discretization to create a set of ordinary differential equations in time to which identification methods for lumped-parameter systems can be applied. This approach, also called time-space separation [9], is the subject of this paper.

There are two recent books related to this paper. The first book by Cressie and Wikle [16] extensively treats statistical modeling and analysis of spatial, temporal, and spatio-temporal data. The second book by Billings [17] addresses spatio-temporal discretized partial differential equations by using polynomial basis functions and model reduction by using orthogonal forward regression.

In this paper, a method for the identification of finite-dimensional models for distributed-parameter systems with a *small number of fixed sensors* is proposed. Compared to other finite-difference identification methods in the literature [17–25], this method does not assume a dense set of measurement locations in space, and, in addition, the method also uses an input selection method to reduce the complexity of the model. The method also allows the use of external inputs in the model, a problem not addressed by Cressie and Wikle [16]. In addition, an application that, to our knowledge, has not yet been described in the literature is presented, namely the identification of a model for the temperature dynamics in a greenhouse.

The remainder of the paper is organized as follows: Section 2 presents the problem formulation for which the method is proposed. Section 3 gives the details of the method. In Section 4, two examples are presented to show the effectiveness of the method: identification using data from a simulation of a 2D heat conduction equation, and identification using temperature measurements of a real-life greenhouse setup. Section 5 concludes the paper.

2. Problem formulation

Consider a distributed-parameter system described by a partial differential equation, with the associated boundary and initial conditions. For the ease of notation and without loss of generality, a system that is first-order in time and second-order in a two-dimensional space is presented:

$$\frac{\partial g(\boldsymbol{z},t)}{t} = f\left(\boldsymbol{z},t,g(\boldsymbol{z},t),\frac{\partial g(\boldsymbol{z},t)}{z_1},\frac{\partial g(\boldsymbol{z},t)}{z_2},\frac{\partial g(\boldsymbol{z},t)}{z_1 z_2},\frac{\partial g(\boldsymbol{z},t)}{z_1^2},\frac{\partial g(\boldsymbol{z},t)}{z_2^2},u(\boldsymbol{z},t),w(\boldsymbol{z},t)\right), \forall \boldsymbol{z}\in\mathcal{Z}\setminus\mathcal{Z}_\mathrm{b}, \forall t \tag{1a}$$

$$0 = h\left(\boldsymbol{z},t,g(\boldsymbol{z},t),\frac{\partial g(\boldsymbol{z},t)}{z_1},\frac{\partial g(\boldsymbol{z},t)}{z_2},u(\boldsymbol{z},t),w(\boldsymbol{z},t)\right), \forall \boldsymbol{z}\in\mathcal{Z}_\mathrm{b}, \forall t \tag{1b}$$

$$g(\boldsymbol{z},t_0) = g_0(\boldsymbol{z}), \forall \boldsymbol{z}\in\mathcal{Z} \tag{1c}$$

Here $g(\cdot,\cdot)$ is the variable of interest, $f(\cdot)$ is the system function, $h(\cdot)$ is the boundary value function, $\boldsymbol{z}=(z_1,z_2)\in\mathcal{Z}\subset\mathbb{R}^2$ is the spatial coordinate,[1] $t\in\mathbb{R}^+\cup\{0\}$ is the continuous-time variable, $u(\cdot,\cdot)$ is the input function, $w(\cdot,\cdot)$ is the process noise, and $\mathcal{Z}_\mathrm{b}$ is the set of spatial boundaries of the system. Higher-order and multi-variable systems can be defined analogously.

Assume that a set of input–output measurements are available from the distributed-parameter system system (1) with unknown functions $f(\cdot)$ and $h(\cdot)$. The sensors are located at specified points to measure $g(\cdot,\cdot)$, and there are also actuators that generate inputs $u(\cdot,\cdot)$ to the system. Since the actuators and the sensors are placed at known and fixed locations, the space is discretized with a set of grid points $\mathcal{M}_\mathrm{g}$ such that the actuator locations $\mathcal{M}_\mathrm{u}$ and the sensor locations $\mathcal{M}_\mathrm{s}$ are in $\mathcal{M}_\mathrm{g}$, i.e., $\mathcal{M}_\mathrm{u}\subset\mathcal{M}_\mathrm{g}$ and $\mathcal{M}_\mathrm{s}\subset\mathcal{M}_\mathrm{g}$. Assume that the measurements, concatenated in a vector $\boldsymbol{y}(\cdot)$, are affected by additive Gaussian noise $v(z_i,t)\sim\mathcal{N}(0,\sigma_{\mathrm{v}_i}^2)$. The input and measurement vectors are defined as:

$$\boldsymbol{u}(t) = \begin{bmatrix} u(\boldsymbol{z}_{\mathrm{u},1},t) & \dots & u(\boldsymbol{z}_{\mathrm{u},N_\mathrm{u}},t)\end{bmatrix}^\top \tag{2a}$$

$$\boldsymbol{y}(t) = \begin{bmatrix} g(\boldsymbol{z}_{\mathrm{g},1},t)+v(\boldsymbol{z}_{\mathrm{g},1},t) & \dots & g(\boldsymbol{z}_{\mathrm{g},N_\mathrm{s}},t)+v(\boldsymbol{z}_{\mathrm{g},N_\mathrm{s}},t)\end{bmatrix}^\top \tag{2b}$$

where N_u is the number of actuators, N_s the number of sensors, the coordinates of the inputs are denoted by $\boldsymbol{z}_{\mathrm{u},j}\in\mathcal{M}_\mathrm{u}$, the measurement coordinates by $\boldsymbol{z}_{\mathrm{g},i}\in\mathcal{M}_\mathrm{g}$, and the superscript $\top$ denotes the transpose of a matrix or vector. Note that not every grid point is associated with a sensor or actuator.

The measurements are collected at discrete time steps $t_k = k\cdot T_s$ with $k\in\mathbb{N}\cup\{0\}$, where T_s is the sampling period. To simplify the notation, the discrete time instant t_k is subsequently written as k. The notation is further simplified by using an integer subscript assigned to the given sensor or actuator location:

$$u_j(k) = u(\boldsymbol{z}_{\mathrm{u},j},t)\big|_{t=k\cdot T_s}, \quad j=1,\dots,N_\mathrm{u} \tag{3}$$

for the inputs and

$$y_i(k) = \left(g(\boldsymbol{z}_{\mathrm{g},i},t)+v(\boldsymbol{z}_{\mathrm{g},i},t)\right)\Big|_{t=k\cdot T_s}, \quad i=1,\dots,N_\mathrm{s} \tag{4}$$

for the outputs. The input and output data (3) and (4) are the only available information to construct a distributed finite-order model of (1).

[1] Vectors are denoted by boldface symbols.

3. Identification method for distributed-parameter systems

The main idea of the method proposed in this paper is to identify at each sensor location a lumped-parameter system, described by a dynamic model. To take into account the spatial dynamics of the system, measurements from the neighboring locations are included as inputs. This is justified by the derivation of coupled discrete-time dynamic models obtained from spatial discretization of a partial-differential equation presented in the beginning of this section. Parameter estimation of the coupled models by solving the least-squares problem is then shown, subsequently followed by model reduction to simplify the models. As measurements and actuation are performed only at some spatial locations, sensors and actuators location related problems are briefly discussed. The summary of the method is given at the end of the section to give big picture of the identification method.

3.1. Construction of coupled discrete-time dynamic models

The discretization of a partial differential equation in space by using the finite-difference method results in a set of coupled ordinary differential equations. At time instant t, the coupling spatially relates the value of the variable of interest at node i, $g_i(t)$, to values of the same variable at the neighboring nodes. The influence of more distant neighbors may be delayed due to the finite speed of spatial propagation of the quantity of interest. As an example, consider the following simplification of (1a) to an autonomous one-dimensional case:

$$\frac{\partial g(z,t)}{t} = m\left(\frac{\partial g(z,t)}{z^2}\right) \tag{5}$$

where $g(z,t) \in \mathbb{R}$ is the variable of interest, $z \in \mathbb{R}$ is the spatial coordinate, and $m(\cdot)$ is a nonlinear function. The system is spatially discretized using the finite-difference method by creating grid points, which, for the sake of simplicity, are uniformly spaced at distance Δ_z. Denote $g_i(t)$ for $g(z, t)$ at grid point $z = i \cdot \Delta_z$, called node i for short. The central approximation [26] of the second-order derivative in space is:

$$\left.\frac{\partial^2 g(z,t)}{\partial z^{2^2}}\right|_{z=i} \approx \frac{g_{i+1}(t) - 2g_i(t) + g_{i-1}(t)}{(\Delta_z)^2} \tag{6}$$

which results in:

$$\frac{dg_i(t)}{t} = m\left(\frac{g_{i+1}(t) - 2g_i(t) + g_{i-1}(t)}{(\Delta_z)^2}\right) \tag{7}$$

Then, using the forward-difference approximation of the time derivative:

$$\left.\frac{dg_i(t)}{t}\right|_{t=k} \approx \frac{g_i(k+1) - g_i(k)}{T_s}$$

to discretize the left-hand side of (7), yields:

$$g_i(k+1) = g_i(k) + T_s \cdot m\left(\frac{g_{i+1}(k) - 2g_i(k) + g_{i-1}(k)}{(\Delta_z)^2}\right) \tag{8}$$

or in a slightly more general form:

$$g_i(k+1) = q\Big(g_i(k), g_{i-1}(k), g_{i+1}(k), T_s, \Delta_z\Big) \tag{9}$$

Note that in this example $g_i(k)$ is influenced only by its immediate neighbors. For systems with a higher spatial order and with exogenous inputs (9) can be written as:

$$g_i(k+1) = q(g_{\mathcal{N}_{s,i}}(k), u_{\mathcal{N}_{u,i}}(k), T_s, \Delta_z) \tag{10}$$

where $g_{\mathcal{N}_{s,i}}(k) = \{g_j(k) | j \in \mathcal{N}_{s,i}\}$ is the set of neighboring variables of interest, including $g_i(k)$ itself and $u_{\mathcal{N}_{u,i}}(k) = \{u_l(k) | l \in \mathcal{N}_{u,i}\}$ is the set of neighboring inputs including $u_i(k)$ itself.

In the system identification setting, Δ_z and T_s are known and fixed and instead of $g_i(k)$ the measurement $y_i(k)$ is used (which includes the effect of measurement noise $v_i(k)$). Thus the following model is obtained:

$$y_i(k+1) = F(y_{\mathcal{N}_{s,i}}(k), u_{\mathcal{N}_{u,i}}(k), v_{\mathcal{N}_{s,i}}(k)) \tag{11}$$

where $y_{\mathcal{N}_{s,i}}(k)$ is the set of neighboring measurements at node i, including $y_i(k)$. The neighbors of node i can be simply the nodes that are within a specified distance ϱ, i.e., $y_{\mathcal{N}_{s,i}}(k) = \Big\{y(\boldsymbol{z}, k) | \, \|\boldsymbol{z} - \boldsymbol{z}_i\| \le \varrho, \boldsymbol{z} \in \mathcal{M}_u \cup \mathcal{M}_s\Big\}$ for measurements and $u_{\mathcal{N}_{s,i}}(k) = \Big\{u(\boldsymbol{z}, k) | \, \|\boldsymbol{z} - \boldsymbol{z}_i\| \le \varrho, \boldsymbol{z} \in \mathcal{M}_u \cup \mathcal{M}_s\Big\}$ for inputs, see Fig. 1. A priori knowledge can be used to obtain a suitable value of ϱ.

An inappropriate choice of ϱ may, however, yield a large number of neighbors that are included in the model. In order to reduce the model complexity, an input or regressor selection method is applied. This topic is discussed later on in Section 3.3.

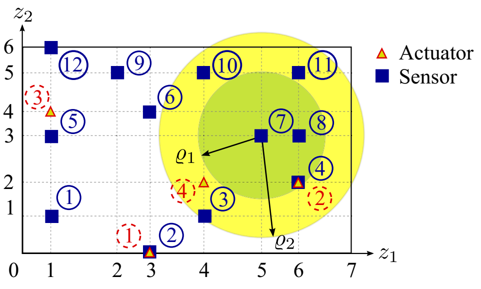


Fig. 1. An illustration of the neighboring measurements and the inputs set with two possible neighborhoods of sensor 7 using a Euclidean distance criterion. The first set of neighbors is defined using distance ϱ_1 from sensor 7 and the second set using distance ϱ_2.

When $F(\cdot)$ in (11) is not known, an approximation can be designed using the available input–output data and linear or nonlinear system identification. Assuming that the system can be approximated by a linear model, linear system identification methods can be applied to (11), as described in the following section.

3.2. System identification and parameter estimation

Identification methods for linear systems (including linear-in-parameters nonlinear systems) use the following model representation:

$$\hat{y}_i(k+1) = \boldsymbol{\phi}_i^\top(k)\boldsymbol{\theta}_i \tag{12}$$

where $\hat{y}_i(k)$ is the predicted value of $y_i(k)$, $\boldsymbol{\phi}_i(k)$ is the regressor vector at time step k, and $\boldsymbol{\theta}_i$ is the vector of parameters. Note that the subscript index i corresponds to sensor i as in the previous section. The regressor vector contains lagged input–output measurements, including those of neighboring sensors and inputs. The parameter vector $\hat{\boldsymbol{\theta}}_i$ can be estimated by using least-squares methods [27], so that the following prediction error is minimized:

$$\hat{\boldsymbol{\theta}}_i = \arg\min_{\boldsymbol{\theta}_i} \sum_{k=1}^{N} \left\| y_i(k+1) - \boldsymbol{\phi}_i^\top(k)\boldsymbol{\theta}_i \right\|_2^2 \tag{13}$$

$$= \arg\min_{\boldsymbol{\theta}_i} \sum_{k=1}^{N} \|\epsilon_i(k)\|_2^2 \tag{14}$$

with $\epsilon_i(k) = y_i(k) - \hat{y}_i(k)$ the prediction error.

The use of neighboring measurements as inputs to the model may lead a situation where the regressors are corrupted by noise. This requires an error-in-variables identification approach, solved, e.g., by using total least squares [28]. For a thorough discussion of the total least-squares method the interested reader is referred to [29]. When noiseless input variables to the actuators are among the regressors, a mixed ordinary-total least-squares method must be used [29]. In this paper, however, the ordinary least squares is used to allow model simplification with methods that extend the ordinary least square, e.g., Lasso. In other words, it is assumed that the prediction error is Gaussian.

In nonlinear system identification, the problem is more difficult as there is no unique way to represent the nonlinear relation between the regressors and the output, and different methods are available to represent the nonlinearity. For instance, Wiener systems [30] and Hammerstein systems [31] use nonlinear functions cascaded with a linear system, Takagi–Sugeno fuzzy models combine local linear models by weighting them via membership functions [32], while neural networks use global nonlinear basis functions [33].

3.3. Model reduction by using regressor selection

Including neighboring measurements as inputs will result in a highly complex model and increase the size of the regressor vector $\boldsymbol{\phi}_i(k)$. This size is determined by the number of neighboring inputs and the number of components of each neighboring regressor vector. A highly complex model may have low generalization performance, i.e., it may not correctly predict previously unseen data. So, the reduction-based approach is used to limit the size of regressor vector and, therefore, only the most relevant components are kept in the model.

The large number of regressors might also cause another problem in case of a limited number of input–output samples, i.e., the regressor matrix has more columns than rows, causing a non-unique least-squares solution. This shows that the least-squares model obtained is ill-posed. Furthermore, simple models are preferred for control applications that use sensors embedded with limited-performance computing devices, i.e., smart sensors. Using simple models reduces the prediction computation load inside the sensors and increases the ease of implementation of the method in real applications. For these reasons, among other reasons, it is desired to have a simpler model by removing inputs that do not contribute to the output to reduce the computational load, especially when the models are used in on-line control design.

Three methods are commonly used to reduce the number of regressors in standard linear regression [34]: stepwise regression, backward elimination, and exhaustive search. With these methods, the inclusion or exclusion of a regressor is decided based on statistical tests, such as the *F*-test.

One of the more recent methods is Lasso [35], which stands for the *least absolute shrinkage and selection operator*. Lasso is a least squares optimization approach with L_1 regularization through a penalty function with an ℓ_1-norm. In Lasso, the following regression model is assumed:

$$\hat{y} = \theta_0 + \boldsymbol{\phi}^\top \boldsymbol{\theta} \tag{15}$$

with $\boldsymbol{\theta} = [\theta_1 \ldots \theta_{n_\mathrm{r}}]^\top$ and θ_0 the parameters of the model and $\boldsymbol{\phi}$ the vector of regressors. Lasso computes the parameters so that the parameters of the regressors that have the least importance are made zero by using a regularization parameter. More specifically, Lasso solves the following optimization problem [35]:

$$\begin{bmatrix} \hat{\theta}_0 & \hat{\boldsymbol{\theta}}^\top \end{bmatrix}^\top = \arg\min_{\theta_0, \boldsymbol{\theta}} \sum_{i=1}^{N} \left(y_i - \theta_0 - \boldsymbol{\phi}_i^\top \boldsymbol{\theta} \right)^2, \quad \text{s.t.} \sum_{j=1}^{n_\mathrm{r}} |\theta_j| \le \tau \tag{16}$$

where τ is a tuning parameter, and for the sake of simplicity the scalar case of y is considered (extension to the vector case is straightforward). This problem can also be written as:

$$\begin{bmatrix} \hat{\theta}_0 & \hat{\boldsymbol{\theta}}^\top \end{bmatrix}^\top = \arg\min_{\theta_0, \boldsymbol{\theta}} \left(\frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \theta_0 - \boldsymbol{\phi}_i^\top \boldsymbol{\theta} \right)^2 + \lambda \sum_{j=1}^{n_\mathrm{r}} |\theta_j| \right) \tag{17}$$

where λ is a nonnegative regularization parameter. Note that formulation (16) and (17) are equivalent in the sense that for any $\tau \ge 0$ in (16), there exists a $\lambda \in [0, \infty)$ in (17) such that the both formulations have the same solution, and vice versa.

As for nonlinear systems there is no unique representation, regressor selection is more complex there. The simplest method, albeit computationally inefficient, is to directly search the most optimal set of regressors using exhaustive search. Regarding model-specific methods, forward regression has been used for polynomial models [36,37], neural networks [38], and for adaptive network fuzzy inference systems [39]. For an example of model-independent regressor selection method, one may refer to, e.g., [40], which uses fuzzy clustering.

3.4. Sensor and actuator locations and interpolation

Measurements and actuations in distributed-parameter systems are commonly performed at spatially sampled locations. This practice raises two related problems in control and estimation of distributed-parameter systems:

1. The number and the locations of sensors and/or actuator. A short introduction to this topic is presented in a survey by Kubrusly and Malebranche [41]. Given the underlying partial differential equations model, the locations of the sensors will influence the identifiability of the distributed-parameter system [42].
   In this paper, the partial differential equations model is assumed unknown and the sensor and actuator locations are assumed to be fixed and given. This resembles the case study of the greenhouse temperature model discussed in Section 4.2. The sensor and actuator location problem is considered a topic of further research.
   Similarly to identification of lumped-parameter systems, it is necessary to generate sufficiently persistent excitation in data acquisition experiments. The notion of persistence of excitation for distributed-parameter systems is more complicated than that for the lumped-parameter systems; see, e.g., [43]. However, this topic is still an open problem and as such, it is out of the scope of the paper.
2. How to interpolate outputs at locations that are not measured? This problem naturally arises because the sensors provide information only at their locations [16].
   For the interpolation problem, kriging and splines are commonly used methods [16]. However, only kriging, more specifically ordinary kriging, is used and briefly presented in this paper following [16]. Kriging was initially developed to solve estimation-related problems in geology and it is able to interpolate in time and space. Because temporal interpolation is not required in our setting, only spatial kriging is given in this section.

Given a spatial random process, also called random field:

$$Y(\boldsymbol{z}) = G(\boldsymbol{z}) + V(\boldsymbol{z}), \quad \boldsymbol{z} \in \mathcal{Z} \tag{18}$$

where $Y(\cdot)$, $G(\cdot)$, and $V(\cdot)$, are respectively the measured random field, the true but unknown random field, and the random measurement noise, $\boldsymbol{z}$ is the spatial coordinate, and $\mathcal{Z}$ is the spatial domain. As the spatial domain $\mathcal{Z}$ has been discretized using the finite-difference method, the measurements of the random field realizations can be written as $y_i = g_i + v_i$, where the subscript i is defined similarly to that of (7), from which the measurement vector $\boldsymbol{y}_\mathrm{z}$ is defined as the stacked measurements from $N_{\boldsymbol{y}_\mathrm{z}}$ sensor locations.

Remarks.

1. Depending on the purpose, a spatio-temporal random process

$$Y(\boldsymbol{z},t) = G(\boldsymbol{z},t) + V(\boldsymbol{z},t), \quad \forall \boldsymbol{z} \in \mathcal{Z},\ \forall t \tag{19}$$

for a certain fixed time t can be viewed as a random field $Y(z)$ or as a dynamic random process $Y(t)$ [44,45]:

$$Y(\boldsymbol{z}) = G(\boldsymbol{z}) + V(\boldsymbol{z}), \quad \forall \boldsymbol{z} \in \mathcal{Z} \tag{20a}$$

$$Y(t) = G(t) + V(t), \quad \forall t \tag{20b}$$

2. In the case of the proposed method, (2b) is the discrete-time realization of (20b) at sensor location $\boldsymbol{z}_i \in \mathcal{M}_\mathrm{S}$.

Kriging [16] is a linear estimation method to obtain the optimal spatial estimate of the second-order stationary process $G(\boldsymbol{z})$ at a coordinate location that is not measured $\boldsymbol{z}_\mathrm{o} \notin \mathcal{M}_\mathrm{S}$, such that the mean square estimation error (MSE):

$$\mathrm{MSE} = \mathbb{E}\left\{\left(g_{\boldsymbol{z}_\mathrm{o}} - \hat{G}(\boldsymbol{y}_\mathrm{z})\right)^2\right\} \tag{21}$$

is minimized, where $g_{\boldsymbol{z}_\mathrm{o}}$ is the true but unknown value of the process $G(\boldsymbol{z})$, $\hat{G}(\boldsymbol{y}_\mathrm{z})$ is the estimator, and $\mathbb{E}\{\cdot\}$ is the expectation operator.

In ordinary kriging, the mean of $G(\boldsymbol{z})$ is assumed constant, i.e., $\mathbb{E}\{G(\boldsymbol{z})\} = \mu_G, \boldsymbol{z} \in \mathcal{Z}$, and the $\mathrm{Cov}(g_i, g_j)$ and the zero mean measurement error variance σ_V^2 are assumed to be known. The estimator has the following form:

$$\hat{G}_\mathrm{O}(\boldsymbol{y}_\mathrm{z}) = \boldsymbol{\gamma}^\top \boldsymbol{y}_\mathrm{z} \tag{22}$$

with the column vector $\boldsymbol{\gamma} \in \mathbb{R}^{N_{\boldsymbol{y}_\mathrm{z}}}$ the estimator parameter. The problem of kriging is to find $\boldsymbol{\gamma}$ to minimize (21). To impose unbiasedness, $\boldsymbol{\gamma}^\top \mathbf{1} = 1$ has to be fulfilled, where $\mathbf{1}$ is a column vector with 1 as the elements. By using the Lagrange multiplier ζ, the parameter vector $\boldsymbol{\gamma}$ is computed by solving the following optimization problem:

$$\arg\min_{\boldsymbol{\gamma}} \left(\mathbb{E}\left\{\left(g_{\boldsymbol{z}_\mathrm{o}} - \boldsymbol{\gamma}^\top \boldsymbol{y}_\mathrm{z}\right)^2\right\} - 2\zeta \cdot (\boldsymbol{\gamma}^\top \mathbf{1} - 1)\right) \tag{23}$$

The solution of the above optimization problem is:

$$\boldsymbol{\gamma}^* = \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1}\left(\mathrm{Cov}\left(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z}\right) + \zeta^* \mathbf{1}\right) \tag{24}$$

$$\zeta^* = \frac{1 - \mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathrm{Cov}\left(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z}\right)}{\mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathbf{1}} \tag{25}$$

where $\boldsymbol{\gamma}^*$ and ζ^* are respectively the optimal parameter vector and Lagrange multiplier, and $\boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}$ is the covariance matrix of measurement vector $\boldsymbol{y}_\mathrm{z}$ defined as:

$$\boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}} = \begin{cases} \mathrm{Var}(y_i) + \sigma_V^2 & i = j \\ \mathrm{Cov}(y_i, y_j) & i \neq j \end{cases} \tag{26}$$

where $\mathrm{Var}(\cdot)$ is the variance. Substituting ζ^* in (24) and $\boldsymbol{\gamma}^*$ into (22) gives:

$$\hat{G}_\mathrm{O}(\boldsymbol{y}_\mathrm{z}) = \left(\mathrm{Cov}(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z}) + \mathbf{1}\frac{1 - \mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathrm{Cov}(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z})}{\mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathbf{1}}\right)^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \boldsymbol{y}_\mathrm{z} \tag{27}$$

with the corresponding mean square error:

$$\mathrm{MSE} = \mathrm{Var}(g_{\boldsymbol{z}_\mathrm{o}}) - \mathrm{Cov}(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z})^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathrm{Cov}(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z}) + \frac{1 - \mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathrm{Cov}(g_{\boldsymbol{z}_\mathrm{o}} \boldsymbol{y}_\mathrm{z})}{\mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathbf{1}} \tag{28}$$

Eq. (27) can be rewritten as:

$$\hat{G}_\mathrm{O}(\boldsymbol{y}_\mathrm{z}) = \hat{\mu}_G + \mathrm{Var}(g_{\boldsymbol{z}_\mathrm{o}})^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \cdot (\boldsymbol{y}_\mathrm{z} - \hat{\mu}_G \mathbf{1}) \tag{29}$$

with $\hat{\mu}_G$ the generalized least-squares estimator of μ_G [46]:

$$\hat{\mu}_G = \frac{\mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \boldsymbol{y}_\mathrm{z}}{\mathbf{1}^\top \boldsymbol{C}_{\boldsymbol{y}_\mathrm{z}}^{-1} \mathbf{1}} \tag{30}$$

Another variant of kriging is universal kriging, which assumes $\mu_G(\boldsymbol{z})$ to be a linear model instead of a constant as in ordinary kriging. An interesting application of this kriging variant is the Kalman filter method for distributed motion coordination strategy of mobile robot positioning at critical locations [47].

3.5. Model validation

Once a model has been built, it needs to be validated [48]. In validation, the model performance is assessed by evaluating its performance to predict data that are not used in the identification, i.e., to predict validation data; to be used for control and estimation purposes. It is desired that the obtained model has a sufficiently good prediction performance, based on a specified measure.

A model is typically presented as a one-step prediction model as shown in (12). In this case the model performance is analyzed by evaluating the errors between the data and its one-step ahead predictions. Larger prediction steps might be required in some applications, e.g., in model predictive control. In this case, the nth step prediction is obtained from

$$\hat{y}_i(k+n) = \hat{\boldsymbol{\phi}}_i(k+n-1)\boldsymbol{\theta}_i^T \tag{31}$$

where $\hat{y}_i(k+n)$ is the predicted value of $y_i(k)$ at discrete time step $k+n$, $\hat{\boldsymbol{\phi}}_i(k+n-1)$ is the regressor vector containing lagged measurements $y_i(k+n-1)$ and/or their predictions $\hat{y}_i(k+n-1)$, and $\boldsymbol{\theta}_i$ is the vector of parameters.

In general, a model with accurate predictions for a long horizon indicates that the behavior of the model is closer to the behavior of the real system. Setting the prediction horizon to infinity represents a very strict test of the model. This is also called the free-run prediction simulation. In free-run simulation, the prediction is computed by using inputs and previous predictions and involving no output measurement.

3.6. Identification procedure

The identification procedure is presented in this section. Given the set of input–output measurements from an unknown distributed-parameter system, the proposed identification procedure proceeds as follows:

1. Create a spatial grid for the system so that each sensor and each actuator is associated with a grid point. The grid may have a uniform or a nonuniform spacing, depending on the actuator and sensor locations. Recall that not all grid points are occupied by sensors or actuators. The sensors and actuators are numbered consecutively: $i = 1, \ldots, N_\mathrm{s}$ for the sensors and $j = 1, \ldots, N_\mathrm{u}$ for the actuators. An illustration of a 2D system, with spatial grid points and labels for the sensors and actuators, is shown in Fig. 2.
2. For each sensor i in the grid:
   (a) Determine the dynamic model structure, using one of the available structures for lumped-parameter systems, such as auto-regressive with exogenous input (ARX), output error (OE), Box–Jenkins (BJ), etc.
   (b) Define the set of neighboring sensors and actuators, i.e., those that are located in a defined neighborhood (details on the notion of neighborhood are given in the next section). The neighboring measurements and inputs from neighboring actuators become inputs to the dynamic model of sensor i. Determine the (temporal) system order and construct the regressors.
   (c) When the number of regressors is large, optimize the model structure in order to simplify the model.
   (d) Estimate the parameters of the dynamic model for sensor i.
   (e) Validate the dynamic model. If the model is rejected, return to step 2(a) to use a different system structure or to 2(b) to change the set of neighbors.

The sequence of the steps and decisions of the method is shown in Fig. 3 and the steps are detailed next. More specifically, it is discussed:

- How to construct coupled discrete-time dynamic models in Section 3.1.
- How to identify and estimate the parameters of the models in Section 3.2.
- How to simplify the identified models to obtain simpler models in Section 3.3.
- Sensor placement and interpolation for locations where measurements are not available in Section 3.4.
- Model validation to assess the performance of models for control or estimation purpose in Section 3.5.

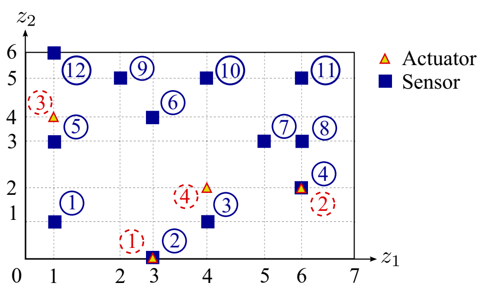


Fig. 2. An illustration of a 2D system with a nonuniform spatial grid. Sensors and actuators are indicated by solid and dashed circles, respectively.

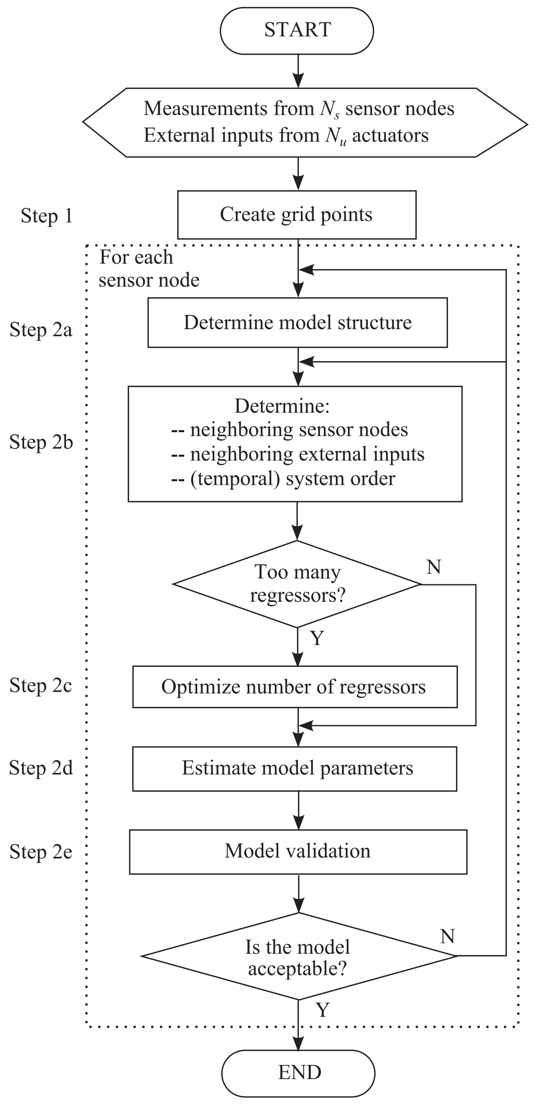

START
Measurements from N_s sensor nodes
External inputs from N_u actuators
Step 1
Create grid points
For each sensor node
Step 2a
Determine model structure
Step 2b
Determine:
-- neighboring sensor nodes
-- neighboring external inputs
-- (temporal) system order
Too many regressors?
N
Y
Step 2c
Optimize number of regressors
Step 2d
Estimate model parameters
Step 2e
Model validation
Is the model acceptable?
N
Y
END

Fig. 3. Flow chart of the proposed method.

Remarks.

- The proposed framework performs off-line identification for distributed-parameter systems; however, the method can be extended directly to recursive identification for the ARX structure.
- For structures that require the predicted output to compute the parameters, extension to recursive parameter estimation is possible provided that the measurements are updated synchronously.
- The convergence for the recursive implementation of the framework can be analyzed by using the methods presented in [48].

4. Simulations and applications

To illustrate the effectiveness of the proposed identification approach, two examples are considered, based on synthetic and real data, respectively. The synthetic data are generated from a linear two-dimensional heat conduction equation. The real-life data are temperature measurements from a small-scale real greenhouse.

In the examples, the model structure selection step is not explicitly presented. This is because the ARX, the firstly tested structure, is already sufficient to obtain acceptable models and no further model structure selection step is required.

Table 1
The plate parameters for the 2D heat conduction equation example.

Parameter	Symbol	Value	Unit
Material density	ρ	4700	$\mathrm{kg\,m^{-3}}$
Thermal conductivity	κ	700	$\mathrm{W\,m^{-1}\,K^{-1}}$
Heat capacity	C_p	383	$\mathrm{J\,kg^{-1}\,K^{-1}}$
Plate length	L	0.7	m
Plate width	W	0.5	m
Initial temperature	T_0	35	°C
Sampling period	T_s	1	s
Grid size	$\Delta_{z_1}, \Delta_{z_2}$	0.05	m

4.1. Heat conduction process

Consider the following two-dimensional heated plate conduction process:

$$\frac{\partial T(\boldsymbol{z},t)}{t} = \frac{\kappa}{\rho C_p}\left[\frac{\partial T(\boldsymbol{z},t)}{z_1^2} + \frac{\partial T(\boldsymbol{z},t)}{z_2^2}\right], \forall \boldsymbol{z} \in \mathcal{Z} \setminus \mathcal{Z}_\mathrm{b}, \forall t \tag{32a}$$

$$T(\boldsymbol{z},t) = T_\mathrm{b}(t), \quad \forall \boldsymbol{z} \in \mathcal{Z}_\mathrm{b}, \forall t \tag{32b}$$

$$T(\boldsymbol{z},0) = T_0, \quad \forall \boldsymbol{z} \in \mathcal{Z} \tag{32c}$$

where $T(\boldsymbol{z}, t)$ is the temperature of the plate at location $\boldsymbol{z}$ and at time t, ρ the density of the plate material, T_0 the initial temperature, C_p the heat capacity, κ the thermal conductivity, and $\boldsymbol{z} = (z_1, z_2)$ the spatial coordinate on the plate. Eqs. (32b) and (32c) are the boundary and initial conditions, respectively. The plate's parameters are listed in Table 1. The values of the material properties are adopted from [49] and modified to speed up the simulation.

For this example, a set of identification data is obtained by simulating the discretized version of (32a). The central approximation of the finite-difference method is used to discretize the space and to create a grid of 14 by 10 cells; the zero-order hold method is used to discretize the time coordinate. The resulting discretized equation is simulated by letting the boundary values $T_\mathrm{b}(\cdot)$ follow pseudo-random binary signals with levels of 25 °C and 80 °C where each boundary B-1 through B-4 (as defined in Fig. 4) is excited by a different signal u_1 through u_4. It is assumed that the excitation is uniformly distributed along the boundary for each discrete-time step k. In case a sensor node has a boundary in the neighborhood, it is taken as one input to the model. The duration of the steps is randomly selected from the set $\{80, 120, \ldots, 200\}$ seconds. The maximum value of the step duration was determined based on the largest time constant of the node responses, i.e., 180 s.

Ten sensor nodes are placed to measure the temperature of the plate as illustrated in Fig. 4, with the exact sensor locations given in Table 2. The measurements are sampled with period $T_s = 1s$ and Gaussian noise with zero mean and variance 0.1 °C^2 is added to the measurements. The data set is divided into an identification set and a validation set, consisting of 1500 and 740 samples, respectively.

The neighboring nodes are defined to be the nodes that lie within the distance $\varrho = 0.35$ m from a given node. The value of this neighborhood radius is set sufficiently large compared to the physical dimensions so that there are sufficient neighboring sensors included in the model. Typically, prior knowledge about the process can be used to determine a suitable value for the radius ϱ.

Results from two representative sensors are presented: 1 and 5. Sensor 1 is relatively close to the boundaries; it has three neighboring sensors. Since boundaries B-1 and B-4 are inside the radius ϱ, the values at boundaries B-1 and B-4 are

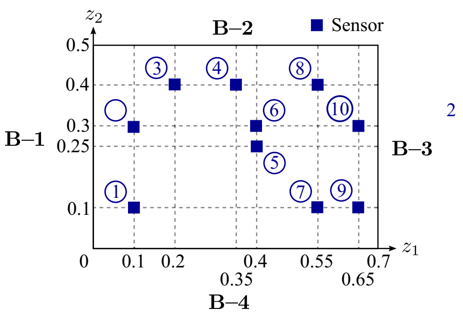


Fig. 4. Illustration of sensor node locations for the 2D heat conduction example.

Table 2
Coordinates of the sensor node locations for the 2D heat conduction equation example.

Sensor #	(z_1, z_2)	Sensor #	(z_1, z_2)
Sensor 1	(0.10, 0.10)	Sensor 6	(0.40, 0.30)
Sensor 2	(0.10, 0.25)	Sensor 7	(0.55, 0.10)
Sensor 3	(0.20, 0.40)	Sensor 8	(0.55, 0.40)
Sensor 4	(0.35, 0.40)	Sensor 9	(0.65, 0.10)
Sensor 5	(0.40, 0.25)	Sensor 10	(0.65, 0.30)

included as inputs to the model of sensor 1. Sensor 5 is near the middle of the plate; it has 9 neighboring sensors and it also uses the values of boundaries B-2–B-4 as inputs.

Subsequently, it is necessary to determine the order of the system. Considering that the system is slow, 10th-order models with an ARX structure are used for the models. Thus, sensor 1 has initially 61 regressors for the model and sensor 5 has 131 regressors including the bias. Lasso is applied to reduce the number of parameters in the model, using the `lasso` function in the Statistics Toolbox of Matlab. The function requires the maximum number of parameters in the model as additional input and returns a set of models with the number of parameters varying from one to the maximum number specified. The function returns a set of reduced models for different values of regularization parameter λ and the corresponding MSE values. Then, one of those models is selected, based on the smallest MSE obtained from the validation data set.

After input selection, a model with 11 parameters is obtained for sensor 1 and a model with 26 parameters for sensor 5. The reduced models are the following:

$$\begin{aligned} y_1(k+1) = {} & 0.0155\,y_1(k-1) + 0.0540\,y_3(k-1) + 0.0467\,y_5(k-1) + 0.4118\,u_1(k-1) + 0.0173\,u_1(k-2) \\ & +0.0026\,u_1(k-3) + 0.4134\,u_4(k-1) + 0.0244\,u_4(k-2) + 0.0034\,u_4(k-3) - 0.5318 \end{aligned} \tag{33}$$

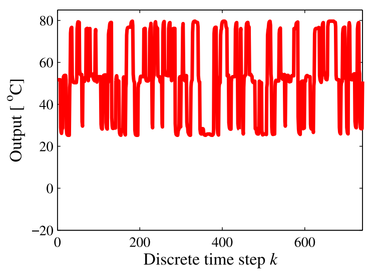
$$\begin{aligned} y_5(k+1) = {} & 0.0089\,y_5(k-1) + 0.0093\,y_8(k-1) + 0.0037\,y_{10}(k-2) + 0.0145\,y_2(k-1) + 0.0050\,y_2(k-2) \\ & +0.0035\,y_2(k-3) + 0.1352\,y_1(k-1) + 0.0241\,y_1(k-2) + 0.0073\,y_1(k-3) + 0.1640\,u_2(k-1) \\ & +0.1074\,u_2(k-2) + 0.0350\,u_2(k-3) + 0.0131\,u_2(k-4) + 0.0016\,u_2(k-5) + 0.0002\,u_2(k-6) \\ & +0.0831\,u_3(k-1) + 0.0627\,u_3(k-2) + 0.0221\,u_3(k-3) + 0.0063\,u_3(k-4) + 0.0006\,u_3(k-5) \\ & +0.1646\,u_4(k-1) + 0.0588\,u_4(k-2) + 0.0245\,u_4(k-3) + 0.0094\,u_4(k-4) \\ & +0.0039u_4(k-5) - 0.8707 \end{aligned} \tag{34}$$

where $y_i(k)$ is the measurement from sensor i, and $u_j(k)$ is the input from boundary j. From the above models, it can be seen that the model for sensor 5 uses more parameters with larger lags of inputs and neighboring measurements; this indicates that more time is needed to propagate those inputs and neighboring measurements to influence sensor 5. This is different in the case of sensor 1, which is closer to the boundaries and for which the resulting model is mainly influenced by the inputs, which yields a simpler model. The models also have constant/bias terms, which can be interpreted as heat transferred between the adjacent nodes.

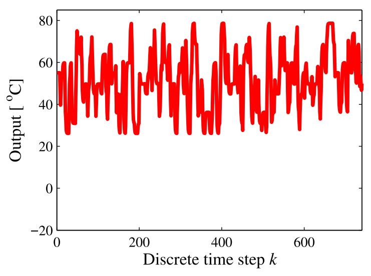
Figs. 5 and 6 show the one-step ahead predictions, the free-run predictions and their corresponding errors in comparison with validation part of the data. As one can expect, for the validation data set the one-step-ahead prediction error is much lower than the error from the free-run simulation. In addition, it can also be seen that the free-run prediction errors are smaller for the reduced input models than those of the full input models. This is more obvious for the model of sensor 5. As one would expect that the full models would deliver smaller errors, this means the full models are suffering from overfitting. In general, the proposed identification approach works well in this case and delivers sufficiently good models.

The figures also show that the output error of the model using measurements from sensor 1 is generally smaller than that of sensor 5. This can be explained as follows: Fig. 4 shows that sensor 5 has more neighboring sensors than sensor 1. This means the identification for measurements of sensor 5 involves more noise from measurements of neighboring sensors than in the case of sensor 1.

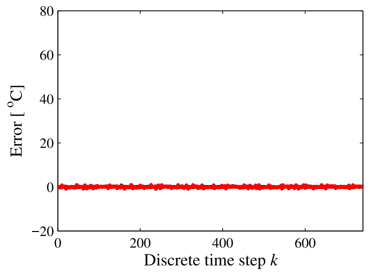
Fig. 7 shows contour plots of the temperature distribution based on the validation data and their one-step-ahead and free-run simulation predictions at discrete-time step $k = 90$; this time step value has been selected in an arbitrary way. The sensor locations are marked with black square boxes where sensor numbers are placed at the left-hand side of the markers. It can be seen the contour of the one-step ahead prediction is very similar to that from the validation data. This is confirmed by the error contour, which is almost uniformly colored around the zero value. Note that the contours look relatively coarse because they are plotted based on sparse measurement locations using ordinary kriging, implemented in the ooDACE toolbox [50,51], to interpolate temperature at locations that are not measured. The R^2 fit for the full models and reduced models of sensor 1 and sensor 5 is shown in Table 3. The table shows that the R^2 fit of the identified models is accurate. It can also be seen for the free-run simulation prediction, the reduced input models have a better R^2 fit than



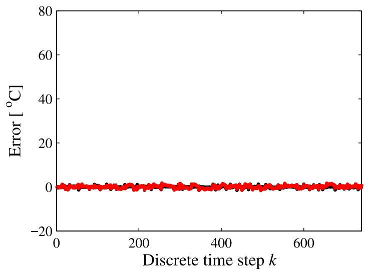
(a) Sensor 1: measurements and predictions



(b) Sensor 5: measurements and predictions

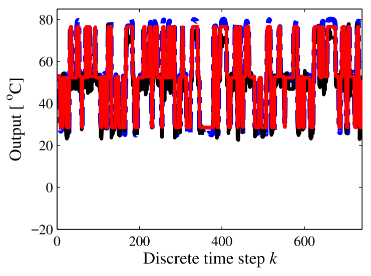


(c) Sensor 1: prediction error

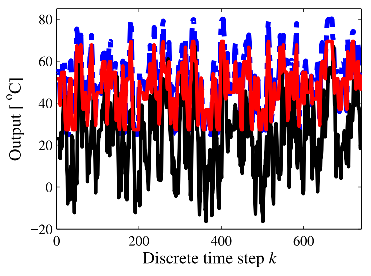


(d) Sensor 5: prediction error

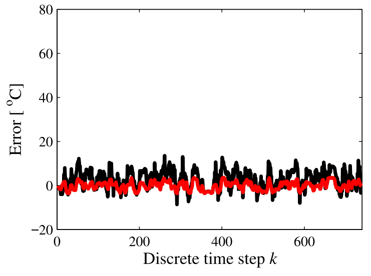
Fig. 5. Measurements (blue, invisible due to the overlap) and one-step ahead prediction for the models with full inputs (black curves) and the ones with reduced inputs (red curves) using the validation data set for the two-dimensional heat conduction example. Note that the prediction error of the full and the reduced input models are overlapping. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



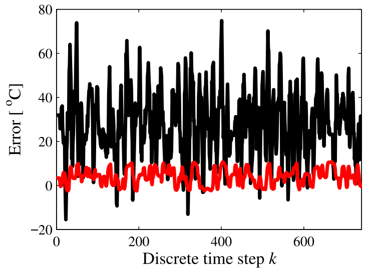
(a) Sensor 1: measurements and simulations



(b) Sensor 5: measurements and simulations



(c) Sensor 1: simulation error



(d) Sensor 5: simulation error

Fig. 6. Measurements (blue) and free-run simulation predictions for the models with full inputs (black) and the ones with reduced inputs (red) using the validation data set for the two-dimensional heat conduction example. Note that the output error of the full and the reduced input models are overlapping. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

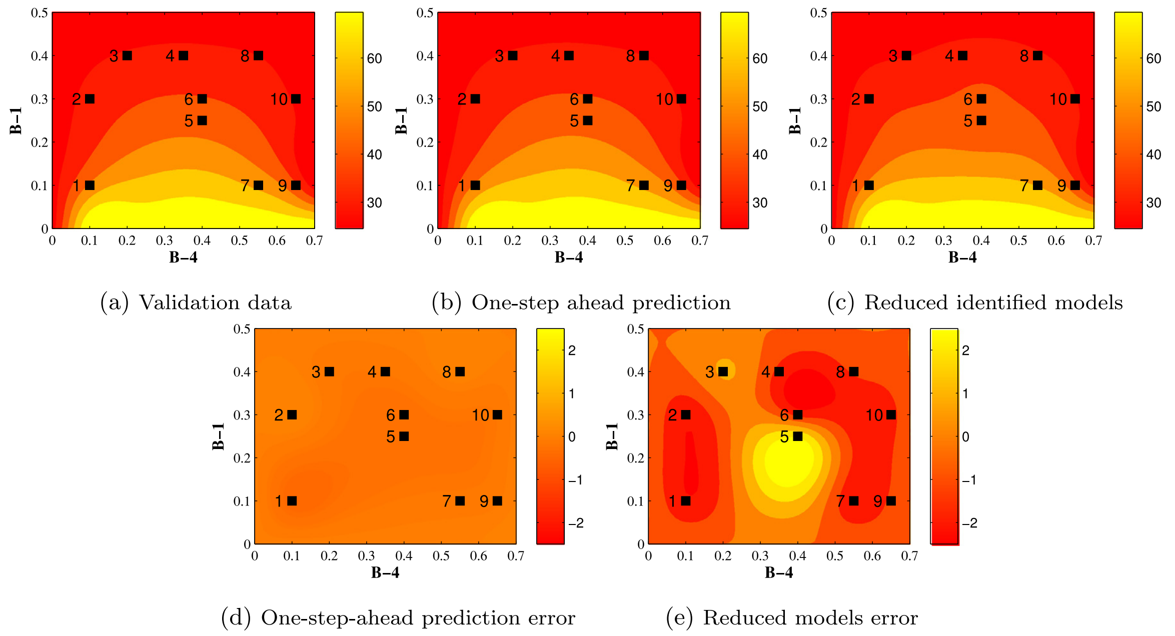


Fig. 7. Contours of the heated plate model at discrete-time step $k = 90$ of the validation data. The black square markers are the sensor locations with their corresponding sensor numbers left of the markers.

Table 3
The R^2 fit of the full and reduced models for one-step ahead and free-run simulation predictions of the heat equation example.

Sensor #	One-step ahead		Free-run simulation	
	Full model (%)	Reduced model (%)	Full model (%)	Reduced model (%)
1	99.9807	99.9784	94.9872	99.0116
5	99.9168	99.8016	−15.5446	93.2703

the full models. This shows that in this case the full models are over-parameterized and that an input reduction results in better models.

Fig. 8 shows the change of the mean square one-step-ahead prediction error. It can be seen that a decrease of the signal-to-noise (SNR) ratio increases the prediction error. The figures also show that the full models have a better prediction performance than the reduced ones, but the difference decreases as the SNR decreases (increase of the noise level). For the full models, the error increases exponentially while for the reduced models it is relatively constant for larger SNR values and increases almost linearly for smaller ones. It can also be seen that for a relatively narrow range of low noise levels, the reduced models exhibit a better robustness than the full ones.

In the numerical examples presented in this paper, the location of the sensors is fixed. However, using the same methodology it is possible to decide on best sensor locations by determining the most appropriate number and locations of the

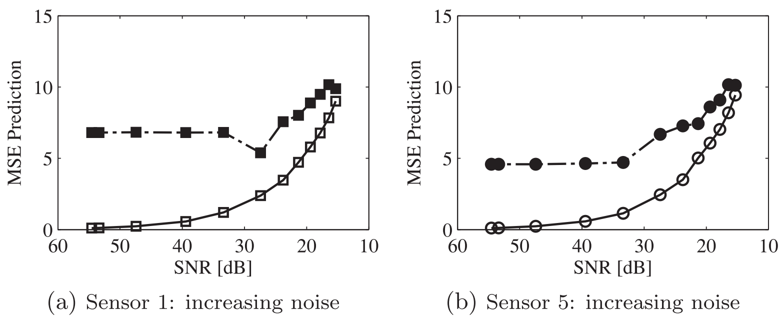


Fig. 8. One-step ahead prediction error of sensors 1 and 5 for the increasing noise variance. The solid lines correspond to the full models and the dashed lines to the reduced models.

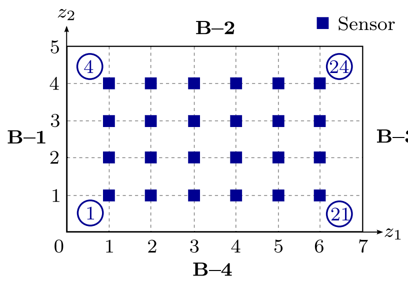


Fig. 9. A grid of sensors for the 2D heat conduction example set up in order to determine appropriate sensor locations.

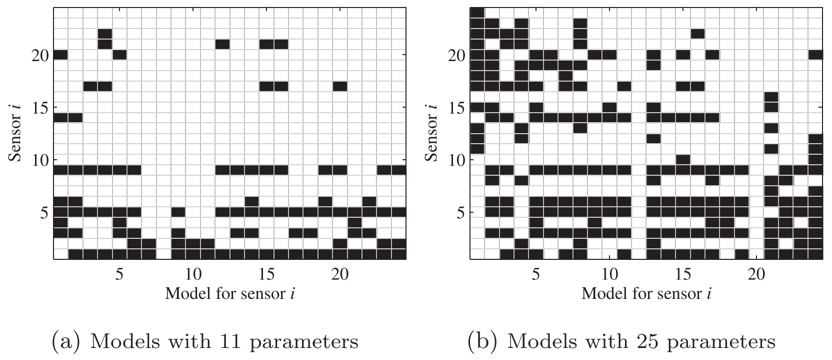


Fig. 10. Sensors whose measurements that are used in the model.

sensors. Assume that the number of sensors is not fixed, but they can be located only within a finite number of possible places. For the 2D heat conduction example, the grid configuration as shown in Fig. 9 represents the potential locations for sensor placement.

The same experiment with the same setup as in the beginning of this section is performed, except that there are now more sensors involved and also the model for measurements from sensor *i* uses measurements from all other sensors as neighbors. After model reduction, we check the regressors from sensor *i* whose non-zero parameters indicate that sensor *i* is used in the model. For the experiment, models with 11 and 25 parameters are selected to be analyzed. The importance of the sensors is represented in the diagrams in Fig. 10. In the figures, the horizontal axis represents the sensor models and the vertical axis represents the sensors involved in the models. The sensors used in a model are indicated by black squares in the vertical direction. If the sensor is not used in the model, a white square is drawn.

A consideration to place a sensor at a certain location is that the measurements from the sensor are used by many models of measurements from the other sensors. The more models use measurements from the sensor, the more important the sensor is. This is indicated by a large number of black squares in the horizontal direction in Fig. 10. In Fig. 10(a), it can be seen that sensors 1, 5, and 9 are important because they are used by the majority of the models. This means that the locations of these sensors are high-potential candidates for the measurement locations. Fig. 10(a) shows several white columns. These columns means that model with the specified number of parameters, in this case 11, cannot be found. The white column locations are different for different numbers of model parameters as shown in Fig. 10(b) for models with 25 parameters. The figure also shows that sensors 3, 6, and 14 become more important by increasing the number of parameters to 25.

4.2. Greenhouse temperature model identification

The proposed approach is also used to identify a model based on data from a small-scale greenhouse setup shown in Fig. 11. The setup was built at TNO in the Netherlands. Its length is 4.6 m, its width 2.4 m, while the height of the wall and the roof are 2.4 m and 2.9 m, respectively. Six 400 W convection heaters, each of 0.6×0.6 m, are placed on the floor of the setup. This gives an average of 200 W m^{-2} irradiance. The heaters are meant to mimic the convective effect caused by the absorption of solar energy by the floor during the day [52]. The coordinates of the centers of the heaters are shown in Table 4. The temperature measurements are collected using wireless sensors, which is a promising technology, with some applications in production greenhouses already reported [53]. For the experiments, a total of 68 sensor nodes have been installed to measure the temperature inside the greenhouse. Out of these, 45 sensor nodes are arranged on a grid with the spacing along the z_1, z_2, and z_3 axes equal to 0.3000 m, 0.7667 m, and 0.5500 m, respectively. Additionally, 5 sensor nodes are placed below the roof, 6 sensor nodes are right at the center of the heaters, and 12 sensors are attached on the four walls of the greenhouse. Fig. 12 shows the sensor locations and Fig. 13 gives a photo of the setup.

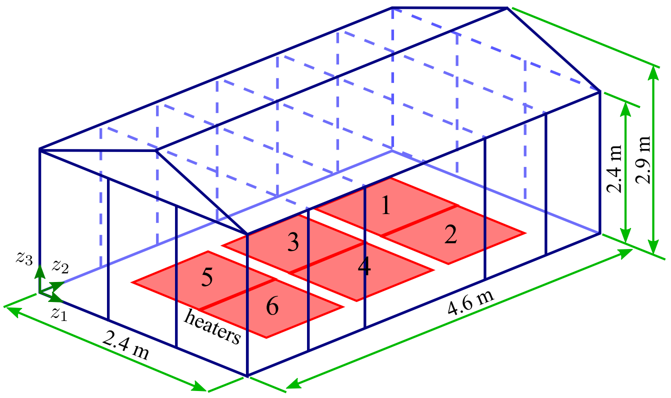


Fig. 11. A schematic representation of the greenhouse with its physical dimensions.

Table 4
The center coordinates of the convection heaters in the greenhouse.

Heater #	(z_1, z_2, z_3)	Heater #	(z_1, z_2, z_3)
Heater 1	(0.90, 3.45, 0.00)	Heater 4	(1.50, 3.45, 0.00)
Heater 2	(0.90, 2.30, 0.00)	Heater 5	(1.50, 2.30, 0.00)
Heater 3	(0.90, 1.15, 0.00)	Heater 6	(1.50, 1.15, 0.00)

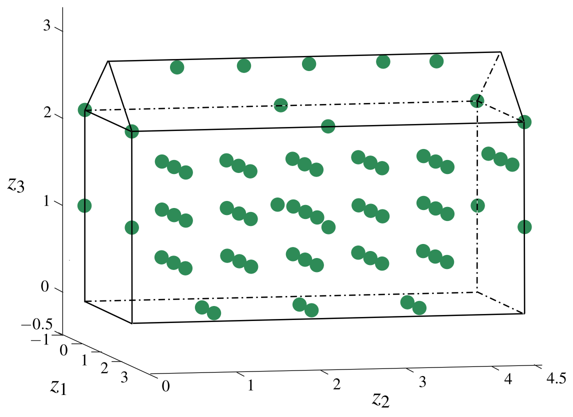


Fig. 12. A schematic representation of the sensor locations in the greenhouse.

Throughout the identification experiments, the heaters were turned on and off in pairs: heater 1 paired with heater 4, heater 2 with heater 5, and heater 3 with heater 6 so that there are three different input signals. In total 3179 data samples have been acquired of which 2149 samples are used for identification and 1030 samples for validation. The data sets are centered by subtracting their means before the identification and model reduction with Lasso are applied.

Among all sensors, identification results from two sensor nodes are presented: sensor node 215, located at position (1.80, 3.83, 1.10) and sensor node 257 located at (0.00, 0.00, 2.20). The neighborhood radius selected is $\varrho = 1.25\,\text{m}$, which gives 19 neighbors for sensor node 215 and 7 neighbors for sensor node 257. Note that the output error of the full and the reduced input models are overlapping.

A 10th-order linear ARX structure is selected for the model so that initially there are 570 parameters and 280 parameters for, respectively, sensor node 215 and 257. The measurements, full input model output, and reduced input model simulation output, as well as the corresponding one-step estimation errors for the validation data, are shown in Fig. 14. Similar plots for 20-step ahead prediction is shown in Fig. 15. Setting the maximum number of parameters to 10, the following models are obtained:

$$\begin{aligned} y_{215}(k+1) = {} & 0.8241\, y_{215}(k-1) + 0.1332\, y_{215}(k-2) + 0.0065\, y_{215}(k-3) + 0.0037\, y_{215}(k-5) \\ & + 0.0041\, y_{215}(k-7) + 0.0043\, y_{215}(k-8) + 0.0113\, y_{206}(k-1) + 0.0048\, y_{218}(k-1) \\ & + 0.0027\, y_{218}(k-2) + 0.0043\, y_{218}(k-3) \end{aligned} \tag{35}$$

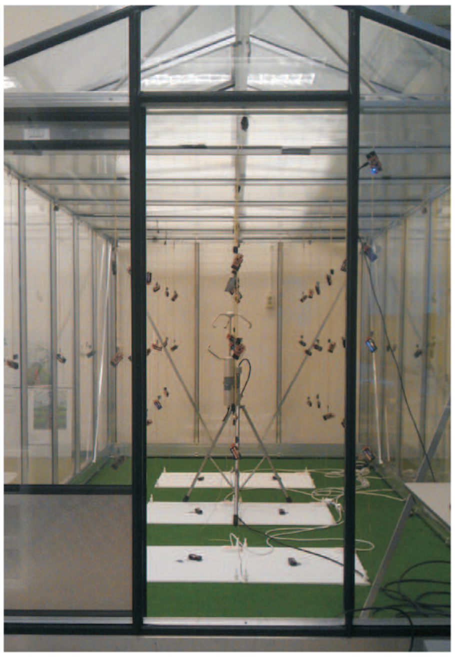


Fig. 13. A photograph of the greenhouse setup used in the case study.

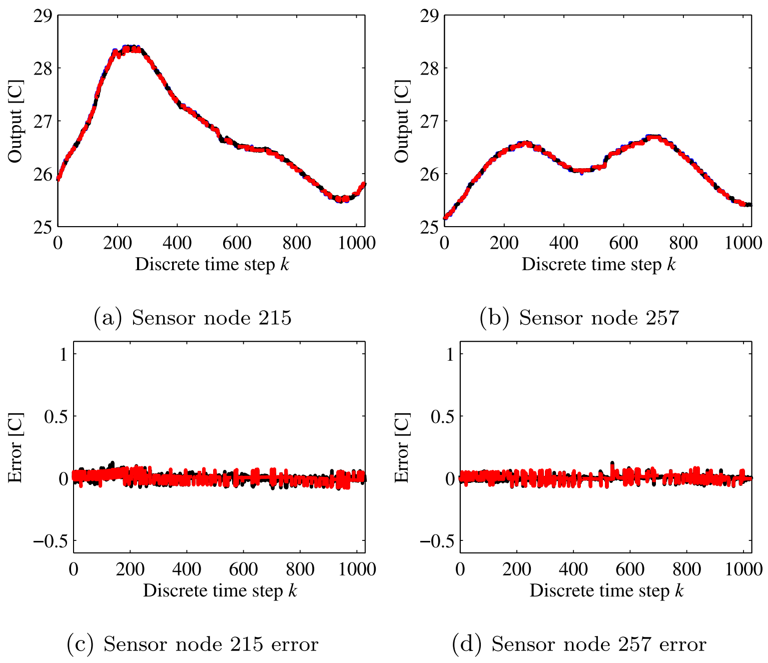


Fig. 14. Greenhouse setup measurements (blue) and one-step-ahead predictions for the model with the full set of inputs (black) and for the model with the reduced set of inputs (red) using the centered validation data set and their corresponding prediction error, i.e., error for the full model (black) and for the reduced model (red). Note that the output and its corresponding error of the full and the reduced input models are overlapping. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$\begin{aligned} y_{257}(k+1) = {} & 0.6206\, y_{257}(k-1) + 0.2410\, y_{257}(k-2) + 0.0093\, y_{257}(k-3) + 0.0368\, y_{257}(k-4) \\ & + 0.0092\, y_{8}(k-1) + 0.0480\, y_{220}(k-1) + 0.0064\, y_{234}(k-1) + 0.0198\, y_{264}(k-1) \\ & + 0.0003\, y_{20}(k-1) + 0.0036\, y_{20}(k-2) \end{aligned} \tag{36}$$

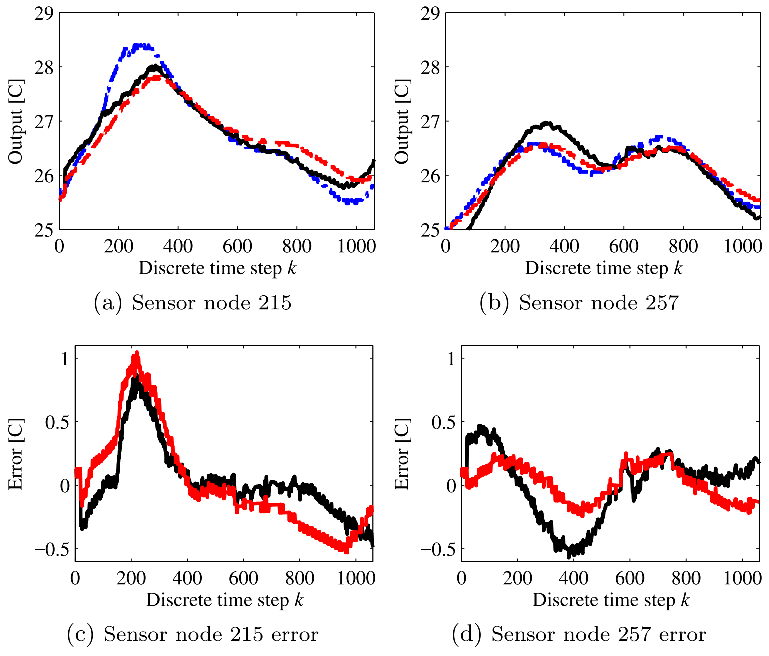


Fig. 15. Greenhouse setup measurements (blue) and 20-step-ahead predictions for the model with the full set of inputs (black) and for the model with the reduced set of inputs (red) using the centered validation data set and their corresponding prediction error, i.e., error for the full model (black) and for the reduced model (red). Note that the measurement, the prediction of the full and the reduced input models are very close one to another. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
Mean square error (MSE) and R^2 fit of the full and reduced models for the greenhouse validation data example in case of one-step and 20-step ahead predictions.

Performance	Sensor 215		Sensor 257	
	Full	Reduced	Full	Reduced
1-step prediction MSE	0.0310	0.0261	0.0229	0.0243
20-step prediction MSE	0.2826	0.3970	0.2600	0.1412
1-step prediction R^2 fit (in %)	99.8772	98.9155	99.6821	99.6515
20-step prediction R^2 fit (in %)	90.0368	80.0343	64.7745	89.8994

where $y_i(k)$ is the measurement at sensor node i. It can be seen that the neighboring measurements contribute to the identified model. The MSE and the R^2 fit for the validation data are shown in Table 5. For sensor 215, it can be seen that the MSE is smaller and the R^2 fit is larger for the reduced input model compared to the full model; while for sensor 257, the MSE increases slightly and the R^2 fit decreases slightly. For the case of sensor 215, the reduction of the R^2 fit suggests the full model is over-parameterized. Increasing the prediction horizon to 20 steps reduces the prediction performance significantly, however the R^2 shows that the models are still stable for prediction up to 20-step ahead. Generally, it can be said that reducing the number of inputs in the models does not significantly decrease the performance of the models. This also indicates that the proposed identification framework works well in this example.

A set of simulations were performed to assess the performance – in terms of the one-step ahead prediction MSE – of the models for different numbers of neighbors for sensor 238. This sensor is located about the middle of the setup and has 8 neighbors with the same height z_3. For neighbor visualization ease, the labeled sensors are shown in Fig. 16. The identification is performed for 2, 4, 6, and 8 neighbors excluding sensor 238 itself. The performance of the full and the reduced models is compared for the validation data. The neighbors and the performance comparison are shown in Table 6. From the table, it can be seen that the one-step ahead prediction errors hardly differs for different numbers of neighbors. This shows the proposed framework is not sensitive to the number of neighboring sensors.

An experiment to estimate values at locations that are not measured is also performed for sensors shown in Fig. 13. In this experiment, data from sensor 217, 238, and 241 are not identified and their estimates for the validation data are obtained by using ordinary kriging. The experiment is performed for both full and reduced models. The kriging models are developed by using the estimates of the validation data of the remaining sensors. The results are shown in Fig. 17 for the estimates and their corresponding error, respectively. The experiment is repeated by omitting sensor 216, 217, 218, 240, 241, and 242. The estimates are shown in Fig. 18 and their corresponding errors in Fig. 19.

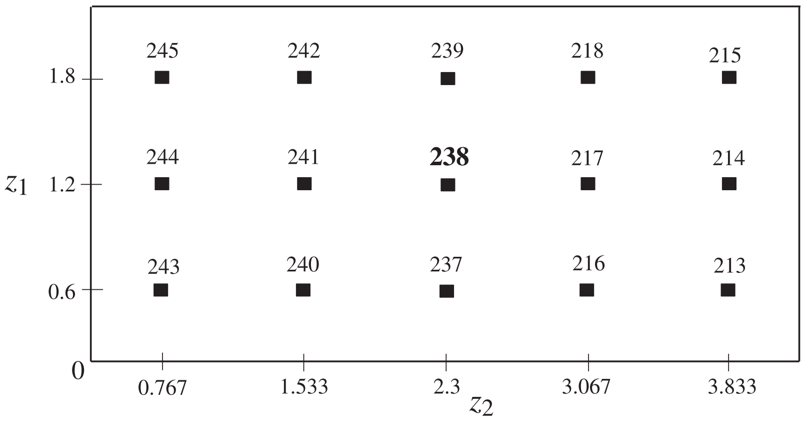


Fig. 16. Sensors at $z_3 = 1.1$ with sensor id labels.

Table 6
The coordinates of sensor 238, its neighbors, and its performance for different numbers of neighboring sensors. The X symbol indicates that the sensor is used as neighbor.

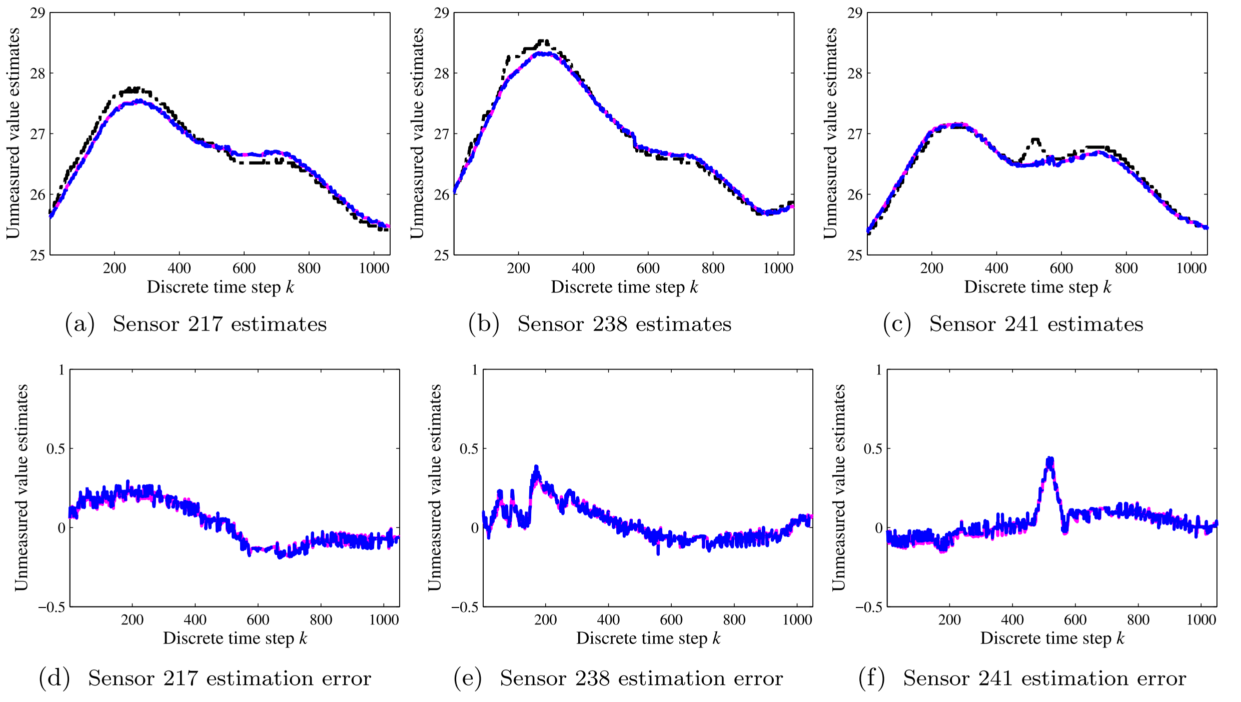
Sensor #	Number of neighbors							
	2		4		6		8	
213		X		X	X		X	
214		X			X		X	X
215				X		X		X
216							X	
217					X		X	X
218						X		X
237						X		
239						X		
240	X		X			X		X
241	X		X	X	X		X	X
242				X	X		X	
243			X			X		X
244			X		X		X	
245							X	X
MSE full	0.0215	0.0215	0.0216	0.0211	0.0220	0.0208	0.0220	0.0222
MSE red	0.0236	0.0230	0.0238	0.0233	0.0235	0.0239	0.0235	0.0239

The figures show that ordinary kriging estimates sufficiently well the values at locations that are not measured. Furthermore, estimation differences between the full and the reduced models are not significant. For the second experiment, it can be seen that the kriging estimates for sensor 216, 217, and 218 look similar; and so are those for sensor 240, 241, and 242. This is can be explained by looking at the validation data from sensor 216, 217, and 218 plotted as a group in Fig. 20(a) and those from sensor 240, 241, and 242 as the other group in Fig. 20(b). From the figure, it can be seen that the temperature difference within a group is small and this creates kriging estimates with insignificant differences among them.

Contour plots of the greenhouse temperature for $0.6 \geq z_1 \geq 1.8$, $0.767 \geq z_2 \geq 3.833$, and fixed $z_3 = 1.1$ are shown in Fig. 21. The plots are in 2D because the ooDACE toolbox is only able to build kriging models from 2D data. In the same way as for the heated plate example, the plots show the contour of the validation data, and the one-step ahead prediction of the full and reduced models. It can be seen that the error is larger with the reduced models than with the full model.

5. Conclusions and future research

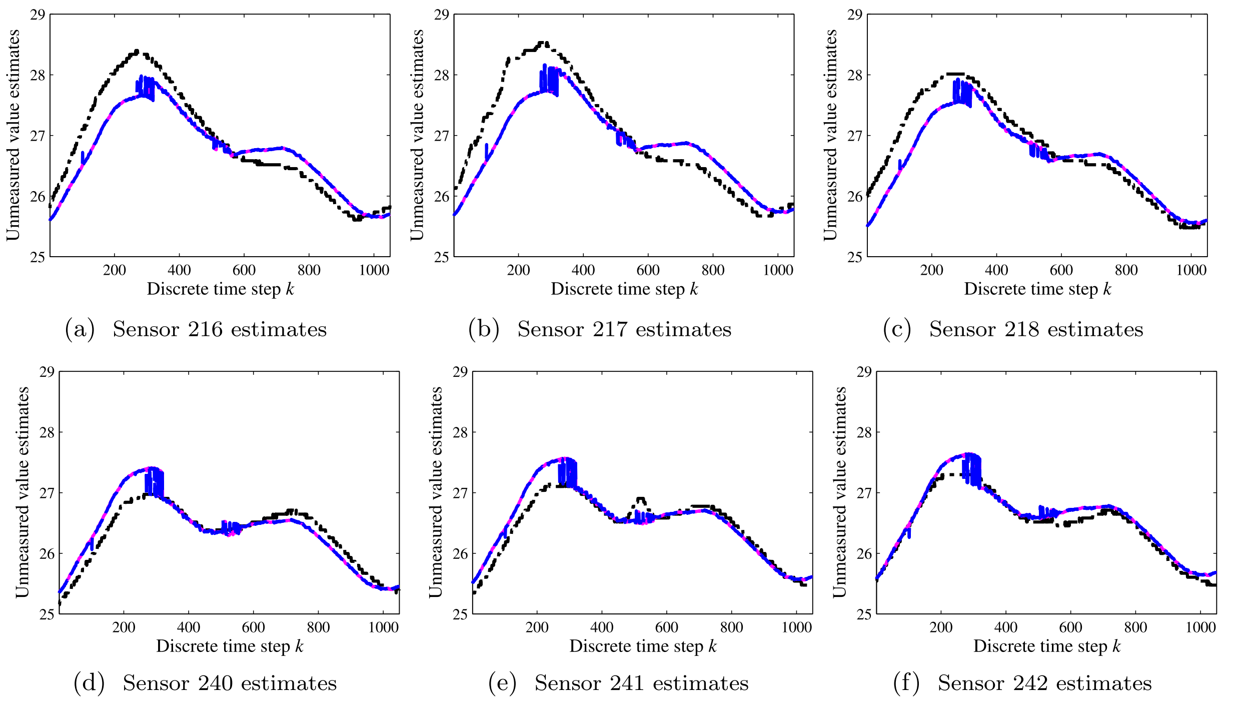
In this paper, a method for identification of distributed-parameter systems was presented. The method is a finite-difference based method that takes into account inputs from neighboring measurements and actuators into the model. The method assumes that the underlying partial differential equation is not known. Although a finite-difference based method is proposed, the method does not require dense measurement locations in the system. This feature allows the applicability of the method to real-life systems, which generally have a limited number of measurements. In addition, model reduction methods were applied to reduce the complexity of the model in case a large number of inputs are involved in the model. The effectiveness of the method has been shown with the help of two examples, a simulated heated plate and a real greenhouse.



(a) Sensor 217 estimates (b) Sensor 238 estimates (c) Sensor 241 estimates

(d) Sensor 217 estimation error (e) Sensor 238 estimation error (f) Sensor 241 estimation error

Fig. 17. Validation data estimates for sensor 217, 238, and 241 by using ordinary kriging and their corresponding error. For (a)–(c), black lines are the validation data, magenta lines are estimates from the full models, and blue lines are estimates from the reduced models. For (d)–(f), magenta lines are errors from the full models and blue lines are errors from the reduced models. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Sensor 216 estimates (b) Sensor 217 estimates (c) Sensor 218 estimates

(d) Sensor 240 estimates (e) Sensor 241 estimates (f) Sensor 242 estimates

Fig. 18. Validation data estimates for sensor 216, 217, 218, 240, 241, and 242 by using ordinary kriging. Black lines are the validation data, magenta lines are estimates from the full models, and blue lines are estimates from the reduced models. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

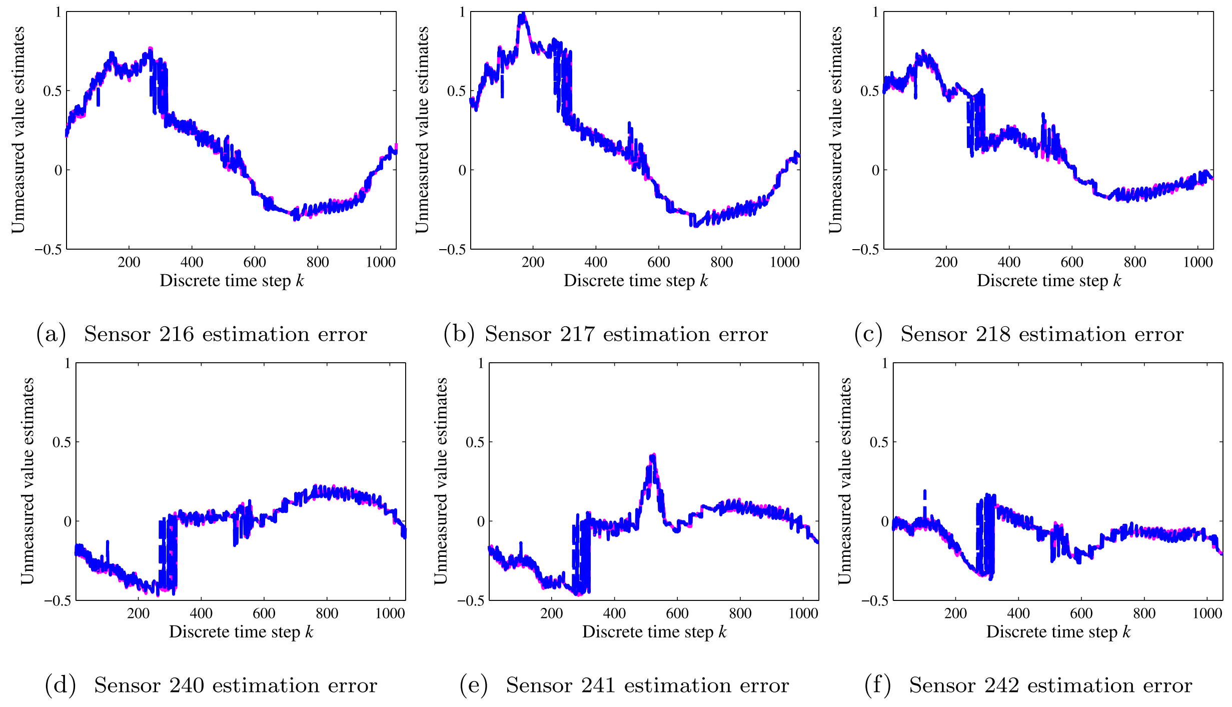


Fig. 19. Validation data estimation error for sensor 216, 217, 218, 240, 241, and 242 by using ordinary kriging. Magenta lines are errors from the full models and blue lines are errors from the reduced models. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

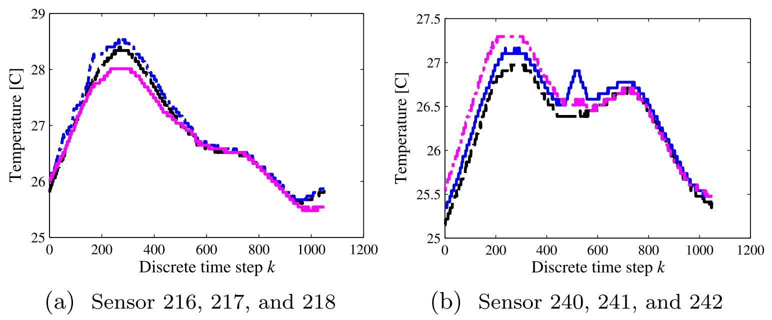


Fig. 20. Validation data plot from sensor: (a) 216 in black, 217 in blue), 218 in magenta (b) 240 in black, 241 in blue, and 242 in magenta. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

There are several open problems related to the proposed method. The first one is how to use the identified model to design a controller or an observer. Models from each sensor can be stacked to form a state space representation, where the measurements at sensor locations represent the states of the system. From the fact that the states are coupled across different measurement locations, the question is how straightforward it is to apply available control design methods for the identified model. The second open problem regards optimal sensor location. In the literature, techniques have been proposed to place sensors for a distributed-parameter system given a certain partial differential model [42]. An extension to handle an unknown or partially known model structure may increase the applicability of the proposed method. The third open problem is the choice of the neighbors. Selecting the right neighbors helps to reduce the computational effort to solve the identification problem. For example, for the greenhouse the neighbor selection is important in case the influence of air flow dynamics inside the modeled chamber cannot be neglected. This example is a limited implementation of the framework currently presented in this paper, the spatial stationary assumption. This leads to the fourth open problem, namely, how to apply the method online in case dynamic neighbor selection is required to handle the air flow dynamics. Finally, further research will focus on the extension of the method to nonlinear distributed-parameter systems.

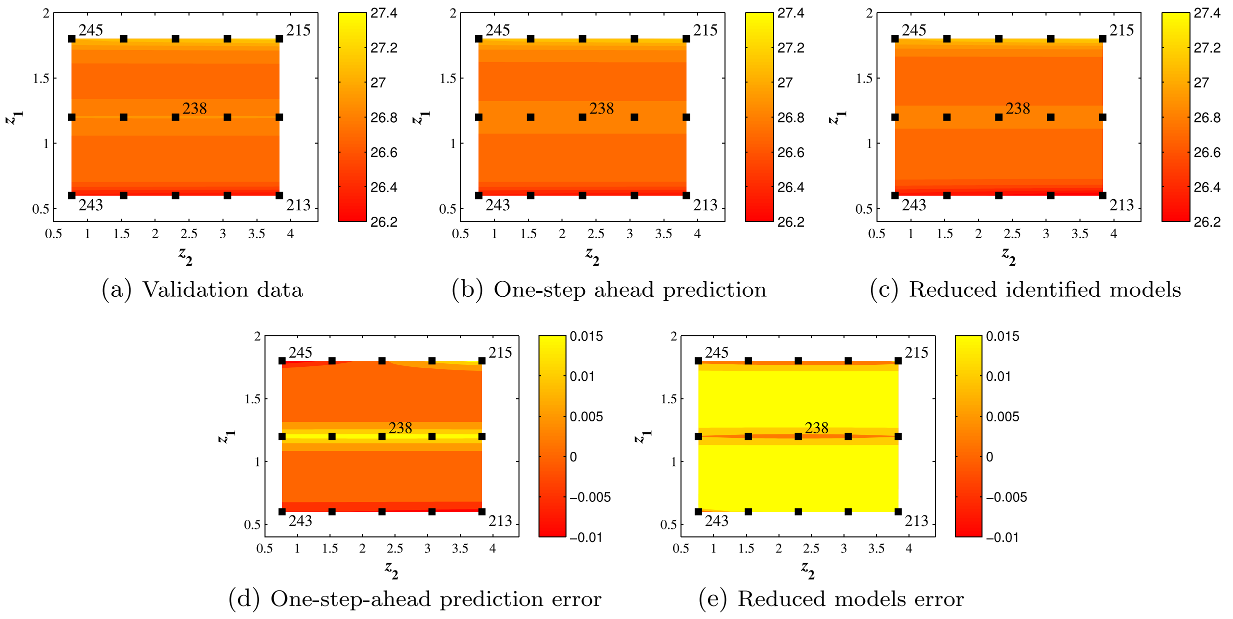


Fig. 21. Contours of the greenhouse temperature model at discrete-time step $k = 400$ of the validation data for $0.6 \geq z_1 \geq 1.8$, $0.767 \geq z_2 \geq 3.833$ and fixed $z_3 = 1.1$. The black square markers are the sensor locations and the labeled sensors are used to build the kriging model.

Acknowledgement

This research was partially funded by the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 257462 HYCON2 Network of Excellence (HYCON2)". The first author gratefully acknowledges the support of the *Government of the Republic of Indonesia, Ministry of Communication and Information Technology*.

References

[1] C. Doumanidis, N. Fourligkas, Temperature distribution control in scanned thermal processing of thin circular parts, IEEE Trans. Control Syst. Technol. 9 (5) (2001) 708–717.
[2] P.D. Christofides, Robust control of parabolic PDE systems, Chem. Eng. Sci. 53 (16) (1998) 2949–2965.
[3] D. Edouard, D. Schweich, H. Hammouri, Observer design for reverse flow reactor, AIChE J. 50 (9) (2004) 2155–2166.
[4] A. Vande Wouwer, C. Renotte, I. Queinnec, P. Bogaerts, Transient analysis of a wastewater treatment biofilter: distributed parameter modelling and state estimation, Math. Comput. Model. Dyn. Syst. Methods Tools Appl. Eng. Relat. Sci. 12 (5) (2006) 423–440.
[5] M. Krstić, R. Vazquez, A.A. Siranosian, M. Bement, Sensing schemes for state estimation in turbulent flows and flexible structures, Proceedings of the SPIE 2006 Smart Structures and Materials 2006: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems vol. 6174 (1) (2006) 61740U.
[6] D. Vries, K.J. Keesman, H. Zwart, A Luenberger observer for an infinite dimensional bilinear system: a UV disinfection example, in: Proceedings of the Third IFAC Symposium on System, Structure and Control, Foz do Iguassu, Brazil, 2007.
[7] R. Klein, N. Chaturvedi, J. Christensen, J. Ahmed, R. Findeisen, A. Kojic, State estimation of a reduced electrochemical model of a lithium-ion battery, in: Proceedings of the 2010 American Control Conference, Baltimore, MD, USA, 2010, pp. 6618–6623.
[8] P. Dufour, P. Laurent, C. Xu, Observer based model predictive control of the water based painting drying using a humidity profile soft sensor and a temperature measurement, in: Proceedings of the First International Workshop and Symposium on Industrial Drying (IWSID), Mumbay, India, 2004. Paper SY152
[9] H.-X. Li, C. Qi, Modeling of distributed parameter systems for applications – a synthesized review from time-space separation, J. Process Control 20 (8) (2010) 891–901.
[10] C. Kravaris, J.H. Seinfeld, Identification of spatially varying parameters in distributed parameter systems by discrete regularization, J. Math. Anal. Appl. 119 (1–2) (1986) 128–152.
[11] J.-L. Lions, Some aspects of modelling problems in distributed parameter systems, in: P.D.A. Ruberti (Ed.), Distributed Parameter Systems: Modelling and Identification, Lecture Notes in Control and Information Sciences, vol. 1, Springer Berlin Heidelberg, 1978, pp. 11–41.
[12] C.S. Kubrusly, Distributed parameter system identification: a survey, Int. J. Control 26 (4) (1977) 509–535.
[13] B. Chen, W. Chen, X. Wei, Characterization of space-dependent thermal conductivity for nonlinear functionally graded materials, Int. J. Heat Mass Transf. 84 (2015) 691–699.
[14] F. Wang, W. Chen, Y. Gu, Boundary element analysis of inverse heat conduction problems in 2D thin-walled structures, Int. J. Heat Mass Transf. 91 (2015) 1001–1009.
[15] B. Chen, W. Chen, A.H.-D. Cheng, L.-L. Sun, X. Wei, H. Peng, Identification of the thermal conductivity coefficients of 3D anisotropic media by the singular boundary method, Int. J. Heat Mass Transf. 100 (2016) 24–33.
[16] N. Cressie, C.K. Wikle, Statistics for Spatio-Temporal Data, John Wiley & Sons, USA, 2011.
[17] S.A. Billings., Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains, John Wiley & Sons, 2013.
[18] S.J. Farlow, The GMDH algorithm of Ivakhnenko, Am. Stat. 35 (4) (1981) 210–215.
[19] H. Park, D. Cho, The use of the Karhunen–Loève decomposition for the modeling of distributed parameter systems, Chem. Eng. Sci. 51 (1) (1996) 81–98.
[20] X.-G. Zhou, L.-H. Liu, Y.-C. Dai, W.-K. Yuan, J. Hudson, Modeling of a fixed-bed reactor using the K–L expansion and neural networks, Chem. Eng. Sci. 51 (10) (1996) 2179–2188.

[21] A. Ben-Nakhi, M.A. Mahmoud, A.M. Mahmoud, Inter-model comparison of CFD and neural network analysis of natural convection heat transfer in a partitioned enclosure, Appl. Math. Modell. 32 (2008) 1834–1847.
[22] D. Coca, S.A. Billings, Direct parameter identification of distributed parameter systems, Int. J. Syst. Sci. 31 (1) (2000) 11–17.
[23] S. Mandelj, I. Grabec, E. Govekar, Statistical approach to modeling of spatiotemporal dynamics, Int. J. Bifurc. Chaos 11 (11) (2001) 2731–2738.
[24] D. Zheng, K.A. Hoo, M.J. Piovoso, Low-order model identification of distributed parameter systems by a combination of singular value decomposition and the Karhunen–Loève expansion, Ind. Eng. Chem. Res. 41 (6) (2002) 1545–1556.
[25] Y. Pan, S.A. Billings, The identification of complex spatiotemporal patterns using coupled map lattice models, Int. J. Bifurc. Chaos (IJBC) Appl. Sci. Eng. 18 (4) (2008) 997–1013.
[26] R.J. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations: Steady State and Time Dependent Problems, SIAM, Philadelphia, PA, USA, 2007.
[27] Å. Björck, Numerical Methods for Least Squares Problems, SIAM, USA, 1996.
[28] T. Söderström, Errors-in-variables methods in system identification, Automatica 43 (6) (2007) 939–958.
[29] S. Van Huffel, J. Vandewalle, The Total Least Squares Problem: Computational Aspects and Analysis, SIAM, Philadelphia, PA, USA, 1991.
[30] L. Zhou, X. Li, F. Pan, Gradient based iterative parameter identification for Wiener nonlinear systems, Appl. Math. Model. 37 (16–17) (2013) 8203–8209.
[31] F. Ding, Hierarchical multi-innovation stochastic gradient algorithm for Hammerstein nonlinear system modeling, Appl. Math. Model. 37 (4) (2013) 1694–1704.
[32] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Syst. Man Cybern. 15 (1) (1985) 116–132.
[33] K. Narendra, K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Netw. 1 (1) (1990) 4–27.
[34] N.R. Draper, H. Smith, Applied Regression Analysis, third ed., Wiley-Interscience, USA, 1998.
[35] R. Tibshirani, Regression shrinkage and selection via the Lasso, J. R. Stat. Soc. Ser. B (Methodol.) 58 (1) (1996) 267–288.
[36] S. Chen, S.A. Billings, W. Luo, Orthogonal least squares methods and their application to non-linear system identification, Int. J. Control 50 (5) (1989) 1873–1896.
[37] E. Mendez, S. Billings, An alternative solution to the model structure selection problem, IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. 31 (6) (2001) 597–608.
[38] G.P. Liu, V. Kadirkamanathan, S.A. Billings, Neural network-based variable structure control for nonlinear discrete systems, Int. J. Syst. Sci. 30 (10) (1999) 1153–1160.
[39] D. Erdirencelebi, S. Yalpir, Adaptive network fuzzy inference system modeling for the input selection and prediction of anaerobic digestion effluent quality, Appl. Math. Model. 35 (8) (2011) 3821–3832.
[40] R. Šindelář, R. Babuška, Input selection for nonlinear regression models, IEEE Trans. Fuzzy Syst. 12 (5) (2004) 688–696.
[41] C. Kubrusly, H. Malebranche, Sensors and controllers location in distributed systems – a survey, Automatica 21 (2) (1985) 117–128.
[42] D. Uciński, Optimal Measurement Methods for Distributed Parameter System Identification, CRC Press, USA, 2004.
[43] Y. Orlov, J. Bentsman, Adaptive distributed parameter systems identification with enforceable identifiability conditions and reduced-order spatial differentiation, IEEE Trans. Autom. Control 45 (2) (2000) 203–216.
[44] J.A. Duan, A.E. Gelfand, C.F. Sirmans, Modeling space-time data using stochastic differential equations, Bayesian Anal. 4 (4) (2009) 733–758.
[45] C.K. Wikle, M.B. Hooten, A general science-based framework for dynamical spatio-temporal models, TEST 19 (3) (2010) 417–451.
[46] A.S. Goldberger, Best linear unbiased prediction in the generalized linear regression model, J. Am. Stat. Assoc. 57 (298) (1962) 369–375.
[47] J. Cortes, Distributed kriged Kalman filter for spatial estimation, IEEE Trans. Autom. Control 54 (12) (2009) 2816–2827.
[48] L. Ljung, System Identification: Theory for the User, second, Prentice Hall, USA, 1999.
[49] J.P. Holman, Heat Transfer, tenth, McGraw-Hill Higher Education, New York, NY, USA, 2009.
[50] I. Couckuyt, F. Declercq, T. Dhaene, H. Rogier, L. Knockaert, Surrogate-based infill optimization applied to electromagnetic problems, Int. J. RF Microw. Comput. Aided Eng. 20 (5) (2010) 492–501.
[51] I. Couckuyt, A. Forrester, D. Gorissen, F. De Turck, T. Dhaene, Blind kriging: implementation and performance analysis, Adv. Eng. Softw. 49 (2012) 1–13.
[52] T. Boulard, G. Papadakis, C. Kittas, M. Mermier, Air flow and associated sensible heat exchanges in a naturally ventilated greenhouse, Agric. Forest Meteorol. 88 (1–4) (1997) 111–119.
[53] N. Wang, N. Zhang, M. Wang, Wireless sensors in agriculture and food industry–recent development and future perspective, Comput. Electron. Agric. 50 (1) (2006) 1–14.