Distributed State Estimation in Large-scale Processes Decomposed into Observable Subsystems Using Community Detection

Leila Samandari Masooleh[1], Jeffrey E. Arbogast[2,3], Warren D. Seider[4], Ulku Oktem[5], and Masoud Soroush[1*]

[1]Department of Chemical and Biological Engineering, Drexel University, Philadelphia, PA 19104, USA

[2]American Air Liquide, Newark, DE 19702, USA

[3]Air Liquide (China) R&D Co., Ltd., Shanghai, China 201108

[4]Department of Chemical and Biomolecular Engineering, University of Pennsylvania, Philadelphia, PA 19104-6393, USA

[5]Near-Miss Management, LLC, 1800 JFK Blvd., Suite 300, Philadelphia, PA 19103, USA

September 9, 2021

REVISED VERSION

Submitted for Publication in *Computers and Chemical Engineering*

Keywords: Nonlinear Kalman filtering, distributed state estimation, network decomposition, observable subsystems, parallel computation

[*]Corresponding author. soroushm@drexel.edu; (215) 895-1710 (phone); (215) 895-5837 (fax)

Abstract

Adequate frequent information on state variables of a process is sometimes needed for effective control and monitoring of the process. However, it is not often available in practice, which can be addressed using a state estimator. This work deals with distributed state estimation in large-scale processes. The decomposition of a process into observable subsystems is formulated as an optimization problem, which is solved using an efficient whale optimization algorithm. Four nonlinear state estimation methods (extended Kalman, unscented Kalman, spherical unscented Kalman, and cubature Kalman filtering) are then implemented and compared using distributed and centralized architectures on a process consisting of two reactors and a separator, and the Tennessee Eastman process. A parallelization strategy that improves the computational efficiency of the distributed architecture is proposed. Simulation results show that the parallel implementation of the distributed filtering methods is computationally more efficient than their centralized counterparts while yielding similarly accurate state estimates.

1. Introduction

Effective control, monitoring, and functional safety of manufacturing processes sometimes require adequate online information on the state variables of the processes[1-5]. However, online measurements of many essential state variables are often not available due to high sensor-hardware costs or the unavailability of reliable online sensors[6]. In such cases, online estimates of unmeasured state variables can be obtained using a state estimator[7], which is model-based and driven by available input and output process measurements. When the process is linear, the classic Kalman filter[8] (KF) or Luenberger observer[9] can be applied. However, linear state estimators are unable to provide accurate state estimates when the process is nonlinear[7]. This inability has motivated the development of nonlinear state estimator design methods[10-12].

Extensive studies on nonlinear filters based on the Bayesian approach have been conducted. The Bayesian approach provides a robust general framework for dynamic state estimation problems. This approach constructs the posterior probability density function (PDF) of the system state based on past and current measurements. The posterior PDF can be obtained using two approximation approaches, local and global. The local approach assumes that the PDF is Gaussian, but the global approach does not. The extended Kalman filtering (EKF)[13], unscented Kalman filtering (UKF)[14], and cubature Kalman filtering (CKF)[15] use the local approach, but particle filtering (PF)[16] is based on the global approach. Many efforts have been made to develop state estimation methods suitable for chemical processes and apply these methods to these processes[5,17-19]. Several recent review articles have put advances in state estimation into perspective; discussed classification, design, and applications of nonlinear state estimators; and highlighted challenges and opportunities in this area [20-23].

A centralized implementation of these nonlinear estimators on large-scale systems is not robust, scalable, or computationally efficient[24]. These limitations can be addressed using a distributed implementation. In a distributed implementation, a large-scale system is decomposed into subsystems with minimum inter-connection and maximum intra-connections. A distributed implementation offers advantages of low computational burden, ease of implementation, and more robustness to sensor failures[25].

Community detection allows for decomposing a system into subsystems with the minimum number of inter-connection and the maximum number of intra-connections. Community detection based on graph theory has attracted a lot of attention in recent years[26,27]. The modularity function

has been used widely as a basis for detecting communities in complex networks[28]. In such approaches, a directed graph is first constructed based on a state-space model. Input, state, and measured variables are treated as nodes and are connected to each other through weighted links. These weighted links represent the strength of the interactions between each pair of nodes. Based on the weighted directed graph, the modularity function is evaluated. Of interest in distributed state estimation (distributed control) is finding the observable (controllable) subsystems that maximize the modularity function. For the purpose of distributed state estimation, Pourkargar *et al*. decomposed a nonlinear system into smaller subsystems using community detection based on a state–output digraph[26,29]. Also, Zhang *et al*. and Yin *et al*. proposed approaches based on weighted graphs and high-gain observers, for distributed state estimation in large-scale processes[30-32].

In this work, we formulate community detection as a multi-objective optimization problem, the solution(s) of which is (are) observable subsystems that maximize the modularity function. A parallel algorithm is proposed and implemented on a computer with parallel processors. Next, a distributed state estimator consisting of a local state estimator for each subsystem is designed. The local estimators communicate to exchange local state estimates and measurements. As each nonlinear filtering method has its own domain of applicability, a single filter may not be optimal or accurate in a wide range of operating conditions. To address this, we investigate distributed implementations of EKF, UKF, spherical unscented Kalman filterings (SUKF), and CKF. The performances of distributed and centralized state estimation schemes are evaluated via numerical simulations. We also develop a parallelized scheme that reduces computation time without sacrificing solution accuracy in the community detection and in the distributed state estimation.

The article proceeds as follows. As preliminaries, general formulations of the KF extensions, the observability concept, and the detailed design of the distributed KF extensions are presented. Next, a method of decomposing a large-scale system into observable subsystems and its parallel implementation are presented. The proposed method is implemented on two case studies with different levels of complexity via numerical simulations, and performances of the four KF extensions are compared when implemented distributedly and centrally. The article ends with concluding remarks.

2. Preliminaries

2.1. Widely-Used Nonlinear State Estimation Methods

2.1.1. Extended Kalman Filtering

An extended Kalman filter (EKF)[13] is designed based on a linear approximation of a nonlinear model around the current estimate. Extended Kalman filtering is the most widely used estimation method due to the ease of its implementation. However, the accuracy of an EKF deteriorates as the degree of nonlinearity of the process increases.

Consider a lumped-parameter system described by a discrete-time state-space model in the form:

$$\begin{cases} x_k = f(x_{k-1}, u_{k-1}) + w_{k-1}\ , & w_{k-1} \sim (0, Q_{k-1}) \\ y_k = h(x_k) + v_k\ , & v_k \sim (0, R_k) \end{cases} \tag{1}$$

where $x \epsilon\ \mathbb{R}^{n_x}$ is the vector of state variables, $y \in \mathbb{R}^{n_y}$ is the vector of output measurements, $u \in \mathbb{R}^{n_u}$ is the vector of input measurements, and $f(.)$ and $h(.)$ are smooth vector functions. w and v are state and output noise vectors, which are assumed to have zero-mean Gaussian distributions with diagonal covariance matrices Q_{k-1} and R_k, respectively. Every process variable is assumed to be normalized (to be dimensionless and vary within [0, 1]). Although inaccurate approximations of process and measurement noise covariances can lead to slower convergence and overall suboptimal estimation performance, it is common to simply provide arbitrary covariance values, as it is often challenging to quantify model and measurement uncertainty in real processes[33]. To improve the robustness of the estimators, process parameters that are highly uncertain are typically identified, modelled as random walk or ramp, and together with the state variables estimated from available measurements. Of course, this combined state and parameter estimation requires a stronger observability condition.

An EKF calculates state estimates using:

$$\hat{x}_k^+ = \hat{x}_k^- + K_k[y_k - h(\hat{x}_k^-)] \tag{2}$$

where

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}$$

$$P_k^- = F_{k-1} P_{k-1}^+ F_{k-1}^T + Q_{k-1}$$

$$\hat{x}_k^- = f(\hat{x}_{k-1}^+, u_{k-1})$$

$$P_k^+ = (I - K_k H_k) P_k^-$$

$$F_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{(\hat{x}_{k-1}^+, u_{k-1})}, \quad H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_k^-}$$

Extended Kalman filtering requires the functions f and h to be differentiable, an initial state estimate ($\hat{x}_0^+$), and an estimate of the estimation error covariance matrix (P_0^+). In the model of Eq.(1), the state and output equations are assumed to be linear in the noise signals. Efforts have been made to apply extended Kalman filtering to the processes in which state and output equations are nonlinear in the noise signals[34]. Several modified extended Kalman filtering techniques, such as the iterated extended Kalman filtering[35] and the second-order extended Kalman filtering[36], have been introduced to improve the performance of EKF. The former improves the linearization error by recursively modifying the center point of the Taylor expansion using $\hat{x}_k^+$ instead of $\hat{x}_k^-$ [35], and the latter reduces the linearization error by considering the second-order term of the Taylor expansion[36].

2.1.2. Unscented Kalman Filtering

Unscented Kalman filtering[14] was proposed as an alternative to extended Kalman filtering. An unscented Kalman filter (UKF) propagates mean and covariance information through nonlinear transformations. A UKF constructs a set of deterministic vectors called sigma points that allow for parameterizing the mean and covariance of a probability distribution. The nonlinear state and output functions are applied to each sigma point to obtain transformed points from which a new mean and covariance estimate are then formed. As a UKF is not based on a linear approximation of a process model, it is suitable for applications in which the model state and output functions are not differentiable. The computational cost of a UKF is comparable to or less than that of an EKF. The greatest advantage of unscented Kalman filtering is that sigma points completely capture the posterior mean and covariance accurately to the third order for any nonlinearity, while EKF will match the mean and covariance up to the first order[34]. Like an EKF, a UKF requires the initial values $\hat{x}_0^+$ and P_0^+.

For a system described by Eq.(1), UKF equations are:

(a)

$$\hat{x}_{k-1}^{(i)} = \hat{x}_{k-1}^{+} + \tilde{x}_{k-1}^{(i)}, \qquad i = 1,,\cdots,2n_x$$

$$\tilde{x}_{k-1}^{(i)} = \left|\left[\sqrt{n_x P_{k-1}^{+}}\right]_i\right|^T, \quad i = 1,,\cdots,n_x$$

$$\tilde{x}_{k-1}^{(i+n_x)} = \left|\left[\sqrt{n_x P_{k-1}^{+}}\right]_i\right|^T, \quad i = 1,,\cdots,n_x \qquad (3)$$

where $[A]_i$ denotes the *i*th row of the matrix A.

(b) Propagating sigma points ($\hat{x}_{k-1}^{(i)}$) through the function f:

$$\hat{x}_k^{(i)} = f(\hat{x}_{k-1}^{(i)}, u_{k-1}) \tag{4}$$

(c) Assigning weight to the propagated sigma points and calculating the prior state estimate and error covariance:

$$\hat{x}_k^- = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\hat{x}_k^{(i)}$$

$$P_k^- = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)\left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)^T + Q_{k-1} \tag{5}$$

(d) Propagating the sigma points ($\hat{x}_k^{(i)}$) through the function h:

$$\hat{y}_k^{(i)} = h\left(\hat{x}_k^{(i)}\right) \tag{6}$$

(e) Assigning weights to the propagated sigma points ($\hat{y}_k^{(i)}$) and calculating an estimate of the output variables at time k:

$$\hat{y}_k = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\hat{y}_k^{(i)} \tag{7}$$

(f) Calculating the prior estimate of the covariance of the output estimates (P_k^y) and the cross-covariance matrix of the state and output estimates (P_k^{xy}):

$$P_k^y = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\left(\hat{y}_k^{(i)} - \hat{y}_k\right)\left(\hat{y}_k^{(i)} - \hat{y}_k\right)^T + R_k$$

$$P_k^{xy} = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)\left(\hat{y}_k^{(i)} - \hat{y}_k\right)^T \tag{8}$$

The term $\frac{1}{2n_x}$ in Eqs.(5), (7), and (8) represents the assigned weight to each sigma point.

(g) Calculating the measurement update using the standard Kalman filter equations:

$$K_k = P_k^{xy}(P_k^y)^{-1}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k[y_k - \hat{y}_k]$$

$$P_k^+ = P_k^- - K_k P_k^y \, {K_k}^T \tag{9}$$

In the preceding UKF algorithm, the state and output equations were assumed to be linear in the noise signals. To handle cases in which the state and output equations are nonlinear in the noise signals, one can use the UKF algorithm in Ref[37].

2.1.3. Spherical Unscented Kalman Filtering

A spherical unscented Kalman filter (SUKF)[38] provides a better sigma point selection strategy by choosing $(n_x + 2)$ sigma points, while keeping the estimation accuracy the same as UKF. Using less sigma points can significantly reduce computational costs. The SUKF algorithm includes the following steps:

(a) Assigning the following scalar weights to the $(n_x + 2)$ sigma points:

$$W^{(0)} \in [0, 1)$$

$$W^{(1)} = \cdots = W^{(n_x+1)} = \frac{1 - W^{(0)}}{n_x + 1} \tag{10}$$

(b) Forming the $n_x \times (n_x + 2)$ matrix:

$$\begin{bmatrix} 0 & \frac{-1}{\sqrt{2W^{(1)}}} & \frac{+1}{\sqrt{2W^{(1)}}} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & \frac{-1}{\sqrt{6W^{(1)}}} & \frac{-1}{\sqrt{6W^{(1)}}} & \frac{2}{\sqrt{6W^{(1)}}} & 0 & \cdots & \cdots & 0 \\ 0 & \frac{-1}{\sqrt{12W^{(1)}}} & \frac{-1}{\sqrt{12W^{(1)}}} & \frac{-1}{\sqrt{12W^{(1)}}} & \frac{3}{\sqrt{12W^{(1)}}} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \frac{-1}{\sqrt{(n_x-1)n_xW^{(1)}}} & \frac{-1}{\sqrt{(n_x-1)n_xW^{(1)}}} & \cdots & \cdots & \cdots & \frac{n_x-1}{\sqrt{(n_x-1)n_xW^{(1)}}} & 0 \\ 0 & \frac{-1}{\sqrt{n_x(n_x+1)W^{(1)}}} & \frac{-1}{\sqrt{n_x(n_x+1)W^{(1)}}} & \cdots & \cdots & \cdots & \frac{-1}{\sqrt{n_x(n_x+1)W^{(1)}}} & \frac{n_x}{\sqrt{n_x(n_x+1)W^{(1)}}} \end{bmatrix}$$

(c) Constructing the sigma points:

$$\hat{x}_{k-1}^{(i)} = \hat{x}_{k-1}^{+} + \sqrt{P_{k-1}^{+}}\,\sigma_i^{(n_x)} \qquad i = 0, 1, \cdots, n_x + 1 \tag{11}$$

where $\sigma_i^{(n_x)}$ is the $(i + 1)$th column of the preceding $n_x \times (n_x + 2)$ matrix.

(d) Propagating the constructed sigma points through the nonlinear state and measurement equations (f and h).

(e) Calculating the prior estimation of the covariance of the output estimates (P_k^y), the cross-covariance matrix of the state and output estimates (P_k^{xy}), and measurement update according to Eqs. (8) and (9).

Since the computational cost in the UKF is proportional to the number of sigma points, SUKF is more attractive in terms of computational costs.

2.1.4. Cubature Kalman Filtering

Cubature Kalman filtering[15] is another Kalman extension that was proposed for high-dimensional state estimation problems. A cubature Kalman filtering (CKF) is a derivative-free estimator and can be applied to those applications for which an analytical from of the Jacobian matrix does not exist. The CKF algorithm uses the cubature rule to solve the multi-dimensional integrals encountered in the nonlinear Bayesian filter[15]. In general, CKF is a special case of UKF for high-dimensional nonlinear filtering problems. The computational burden of CKF is similar to the UKF with better numerical stability[15].

Assuming that at time k the PDF is $p(x_{k-1}|D_{k-1}) = N(\hat{x}_{k-1}^+, P_{k-1}^+)$, the CKF algorithm can be summarized as follows:

(a) Calculating cubature points ($\hat{x}_{k-1}^{(i)}$) for $i = 1,2,\cdots,2n_x$, and propagating them through the function f:

$$\hat{x}_{k-1}^{(i)} = \sqrt{P_{k-1}^+}\xi_i + \hat{x}_{k-1}^+$$
$$\hat{x}_k^{(i)} = f\left(\hat{x}_{k-1}^{(i)}, u_{k-1}\right) \tag{12}$$

where ξ_i is the unit cubature points and is defined as $\sqrt{n_x}\{1\}_i$, $\{1\}_i$ is the ith column of matrix $[\, I_{n_x} \; -I_{n_x}]$, and n_x is the identity matrix of size n_x.

(b) Estimating the predicted state and error covariance:

$$\hat{x}_k^- = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\hat{x}_k^{(i)}$$
$$P_k^- = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)\left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)^T + Q_{k-1} \tag{13}$$

(c) Evaluating cubature points ($\hat{x}_k^{(i)}$), for $i = 1,2,\cdots,2n_x$ and propagating them through the function h:

$$\hat{x}_k^{(i)} = \sqrt{P_k^-}\xi_i + \hat{x}_k^-$$
$$\hat{y}_k^{(i)} = h\left(\hat{x}_k^{(i)}\right) \tag{14}$$

(d) Estimating the predicted measurement, measurement covariance, and cross-covariance matrix using:

$$\hat{y}_k = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\hat{y}_k^{(i)}$$

$$P_k^y = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\left(\hat{y}_k^{(i)}\right)\left(\hat{y}_k^{(i)}\right)^T - (\hat{y}_k)(\hat{y}_k)^T + R_k$$

$$P_k^{xy} = \frac{1}{2n_x}\sum_{i=1}^{2n_x}\left(\hat{x}_k^{(i)}\right)\left(\hat{y}_k^{(i)}\right)^T - (\hat{x}_k^-)(\hat{y}_k)^T \tag{15}$$

(e) Estimating Kalman gain, update state, and error covariance using:

$$K_k = P_k^{xy}(P_k^y)^{-1}$$
$$\hat{x}_k^+ = \hat{x}_k^- + K_k[y_k - \hat{y}_k]$$
$$P_k^+ = P_k^- - K_k P_k^y\ {K_k}^T \tag{16}$$

2.2. Observability of nonlinear systems

Observability is a major requirement in the design of state estimators for dynamic systems. Its existence indicates that output measurements contain information on all state variables[7]. To implement distributed state estimation, we decompose an observable system into a set of observable subsystems. The system of Eq.(1) is locally observable if the following $n_x n_y \times n_x$ observability matrix is full column rank:

$$\frac{\partial}{\partial x}\begin{bmatrix} y(k) \\ y(k+1) \\ \vdots \\ y(k+n_x-1) \end{bmatrix}_{(x_{ss},u_{ss})} = \frac{\partial}{\partial x}\begin{bmatrix} h(x) \\ h\circ f(x,u) \\ \vdots \\ h\underbrace{\circ f\circ\cdots\circ f(x,u)}_{f \text{ is repeated } (n_x-1) \text{ times}} \end{bmatrix}_{(x_{ss},u_{ss})}$$

where $h \circ f(x,u) = h\big(f(x,u)\big)$, $h \circ f \circ f(x,u) = h\big(f(f(x,u),u)\big)$, and so on. u_{ss} and x_{ss} are the steady-state values of the input and state vectors. Alternatively, one can evaluate the structural observability of a system by determining whether all state variables affect measured output(s) directly or indirectly. In the next section, we propose a systematic approach for checking the structural observability of a system. The approach is then used to ensure that each subsystem is observable from its local measurements.

2.3. Distributed State Estimation Scheme

The majority of previous studies on distributed state estimation have focused on a given distributed architecture [39,40]. The literature on distributed state estimation includes approaches based on Kalman filtering [41], particle filtering [42], and moving-horizon estimation [43] and others. Since most of these approaches have addressed challenges like preserving the stability, performance, and robustness as much as their centralized counterparts, it is sufficient to review the concept of distributed schemes.

A distributed state estimation architecture for a large-scale system is depicted in Figure 1. For each subsystem, a local estimator is designed. The local estimators exchange their local input and output measurements as well as their state estimates over the network to exchange information. At each time instant k, an ith local estimator calculates the state variable estimates $\hat{x}_k^{(i)}$ in parallel, by employing the local input and output measurements, $u_k^{(i)}$ and $y_k^{(i)}$, as well as the past estimates of the state variables of all subsystems received by communicating over the network.

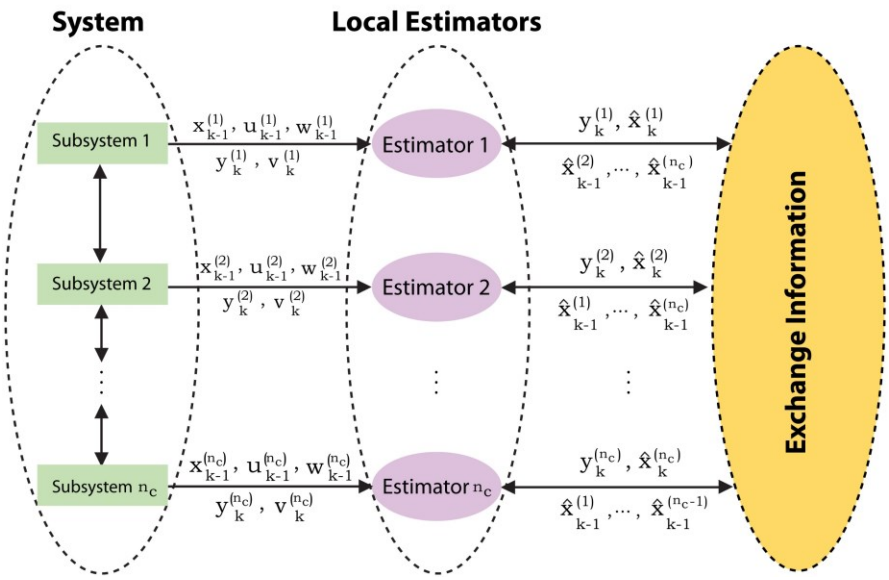

Figure 1: Distributed state estimation.

3. Decomposing a Nonlinear Complex System into Observable Subsystems

3.1. Optimization Formulation

We optimally decompose an observable nonlinear system in the form of Eq.(1) into the n_C observable subsystems:

$$\begin{cases} x_k^{(i)} = f^{(i)}\left(x_{k-1}^{(i)}, u_{k-1}^{(i)}, \emptyset_{k-1}^{(i)}\right) + w_{k-1}^{(i)} \\ y_k^{(i)} = h^{(i)}\left(x_k^{(i)}\right) + v_k^{(i)} \end{cases} \qquad i = 1, \cdots, n_C \tag{17}$$

where $x^{(i)} \in \mathbb{R}^{n_{x_i}}$, $y^{(i)} \in \mathbb{R}^{n_{y_i}}$, and $u^{(i)} \in \mathbb{R}^{n_{u_i}}$ is the vectors of process state variables, measured outputs, and measured inputs of the ith subsystem, and $\emptyset^{(i)}$ is the vector of the state variables of the remaining $(n_C - 1)$ subsystems.

The basis for the decomposition is the degrees of interactions among the state variables, input variables, and outputs. A measure of each interaction is the sensitivity of one variable to another:

$$\tilde{S}_{ij} = \left.\frac{\partial f_i}{\partial u_j}\right|_{ss}, \quad \bar{S}_{ij} = \left.\frac{\partial f_i}{\partial x_j}\right|_{ss}, \quad \bar{\bar{S}}_{ij} = \left.\frac{\partial h_i}{\partial x_j}\right|_{ss} \tag{18}$$

where $\tilde{S}_{ij}$ is a measure of the sensitivity of x_i to u_j, $\bar{S}_{ij}$ a measure of the sensitivity of x_i to x_j, and $\bar{\bar{S}}_{ij}$ a measure of the sensitivity of y_i to x_j. The results of such a sensitivity analysis can be presented in matrix form.

$$
[S_{ij}] = \begin{array}{c}
\begin{matrix} u_1 & \cdots & u_{n_u} & x_1 & \cdots & x_{n_x} & y_1 & \dots & y_{n_y} \end{matrix} \\
\left[\begin{array}{ccccccccc}
0 & \cdots & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\
\tilde{S}_{11} & \cdots & \tilde{S}_{1n_u} & \bar{S}_{11} & \cdots & \bar{S}_{1n_x} & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
\tilde{S}_{n_x1} & \cdots & \tilde{S}_{n_xn_u} & \bar{S}_{n_x1} & \cdots & \bar{S}_{n_xn_x} & 0 & \cdots & 0 \\
0 & \cdots & 0 & \bar{\bar{S}}_{11} & \cdots & \bar{\bar{S}}_{1n_x} & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & \bar{\bar{S}}_{n_y1} & \dots & \bar{\bar{S}}_{n_yn_x} & 0 & \cdots & 0
\end{array}\right]
\begin{matrix} u_1 \\ \vdots \\ u_{n_u} \\ x_1 \\ \vdots \\ x_{n_x} \\ y_1 \\ \vdots \\ y_{n_y} \end{matrix}
\end{array}
$$

In this work, we use the relation:

$$
w_{ij} = \begin{cases} \left|(\log|S_{ij}|)\right|, & S_{ij} \neq 0 \\ 0, & S_{ij} = 0 \end{cases}, \qquad i,j = 1, \cdots, (n_x + n_u + n_y)
$$

to assign a weight to each pair of the interactions.

Based on the degrees of the interactions (weights), a modularity index is then defined[28]:

$$
Q_w = Q_1 - Q_2 \tag{19}
$$

where

$$
Q_1(C_1, \cdots, C_{n_C}) = \frac{1}{W} \sum_i \sum_j w_{ij}\ \delta(M_i, M_j),
$$

$$
Q_2(C_1, \cdots, C_{n_C}) = \frac{1}{W} \sum_i \sum_j \frac{w_i^{out} w_j^{in}}{W}\ \delta(M_i, M_j)
$$

where $w_i^{out} = \sum_j w_{ji}$, $w_j^{in} = \sum_i w_{ji}$, and $2w = \sum_i w_i^{out} = \sum_j w_j^{in} = \sum_i \sum_j w_{ij}$. $C_1, \cdots, C_{n_C}$ are the sets of the variables (nodes) that the communities (subsystems) $1, \cdots, n_C$ include, respectively. M_i is the ith node, and δ is the Kronecker delta symbol; if both M_i and M_j belong to the same community, $\delta(M_i, M_j)$ is equal to 1; otherwise, it is zero:

$$\delta(M_i, M_j) = 1, \qquad \text{if } M_i, M_j \in C_l$$
$$\delta(M_i, M_j) = 0, \qquad \text{if } M_i \in C_l, M_j \notin C_l$$

We require each subsystem to be structurally observable, as defined in the next definition. The use of structural observability allows one not to consider weak connections among variables and prevents the algorithm from finding structures that have an ill-conditioned observability matrix. Of course, the user decides on the interaction-strength threshold; interactions, the strengths of which are below this threshold are ignored.

Definition 1. A system in the form of Eq.(1) is structurally observable if the following $n_x n_y \times n_x$ matrix is full column rank:

$$\begin{bmatrix} \bar{\bar{D}} \\ \bar{\bar{D}}\bar{D} \\ \vdots \\ \bar{\bar{D}}\bar{D}^{n_x-1} \end{bmatrix}$$

where $\bar{D} = [\bar{D}_{ij}]$, $\bar{\bar{D}} = [\bar{\bar{D}}_{ij}]$, and

$$\bar{D}_{ij} = \begin{cases} 1, & |\bar{S}_{ij}| \geq \epsilon_1 \\ 0, & |\bar{S}_{ij}| < \epsilon_1 \end{cases}, \qquad i, j = 1, \cdots, n_x$$

$$\bar{\bar{D}}_{ij} = \begin{cases} 1, & |\bar{\bar{S}}_{ij}| \geq \epsilon_2 \\ 0, & |\bar{\bar{S}}_{ij}| < \epsilon_2, \end{cases} \quad \begin{matrix} i = 1, \cdots, n_y \\ j = 1, \cdots, n_x \end{matrix}$$

Here, ϵ_1 and ϵ_2 are positive scalar constants are set by the user.

Definition 2. If a system in the form of Eq.(1) is structurally observable in the sense of Definition 1, then the structural observability index of the system, $O = 1$, otherwise $O = 0$.

We formulate community detection as a multi-objective optimization problem. Multi-objective community detection methods describe multiple structure properties of networks by optimizing two conflicting objectives, intra-connections (Q_1) and inter-connections (Q_2).

Using these definitions, the resulting community detection problem is:

$$\max_{n_C, C_1, \cdots, C_{n_C}} Q_w\,(C_1, \cdots, C_{n_C}) = \gamma Q_1(C_1, \cdots, C_{n_C}) \; - (1-\gamma) Q_2(C_1, \cdots, C_{n_C}) \tag{20}$$

subject to:

$$O_1 = \cdots = O_{n_C} = 1$$

where $\gamma \in [0, 1]$ and is varied in this range to find the Pareto front.

The objective of the optimization problem here is to maximize Q_w to identify the communities that are structural observable. Since in the described multi-objective optimization problem, there does not typically exist a feasible solution that maximizes Q_1 and minimizes Q_2 simultaneously, we use the concept of the Pareto optimal to find the set of non-dominated solutions.

The optimization problem defined in Eq.(20) is also subject to the observability of all produced subsystems since the observability of the entire system does not guarantee the observability of produced subsystems. These constraints can be relaxed by requiring the detectability of every subsystem[7].

In the set of Pareto observable optimal solutions, the configuration corresponds to the largest modularity is preferred, as a larger value of modularity indicates better partitioning. Moreover, among Pareto observable optimal solutions, those solutions aligning with the physical topology of the system are more desirable than any other solution. Because this approach benefits the reduction of the computational burden associated with their implementation in real-time and the ease of use in distributed state estimation implementation.

3.2. A Multi-objective WOA Algorithm for Community Detection in Large-scale Systems

Since community detection based on optimizing a modularity function is an NP-hard problem[44], metaheuristic algorithms have been adopted to solve NP-problems owing to their simplicity, ease of implementation, and the ability to avoid local optima.[45]

It has been demonstrated by Masooleh et al.[46] that solving the modularity function using WOA[47] results in finding solutions in a short computing time. The whale optimization algorithm is a metaheuristic algorithm that imitates the social behavior of humpback whales. In this algorithm, the bubble-net hunting strategy of humpback whales is exploited. However, this algorithm, in its present form, is appropriate for solving single-objective optimization problems with continuous variables. To make it applicable to solve multi-objective optimization problems with discrete variables, a discrete version of this algorithm was proposed[46]. In the proposed approach, a transfer function is utilized to update the position of the whales. A non-sorting genetic algorithm (NSGA)-II[48] has been used to generate a list of non-dominated solutions. In the non-dominated sorting method, a comparison between each solution with every single solution is made

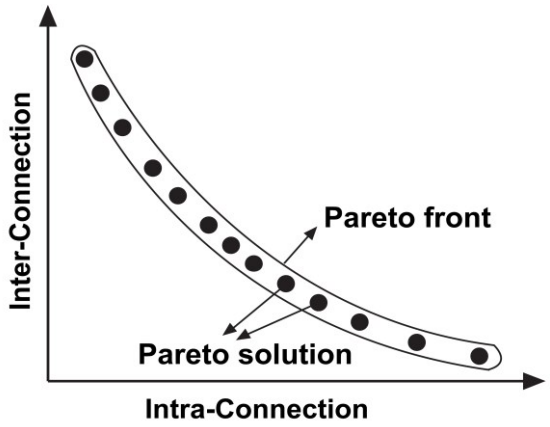

Figure 2: Graphical illustration of Pareto front and Pareto optimal solutions in community detection problems.

to check whether the solutions dominate each other. This comparison is made for all individuals to find Pareto optimal members (Rank #1). Figure 2 shows the graphical illustration of the Pareto front and Pareto optimal solutions in community detection problems.

Apart from convergence to the Pareto-optimal set, the solutions should be diverse along the Pareto front. The crowding distance mechanism is employed to preserve the diversity along the Pareto front[46,48]. The crowding distance is calculated in the same front to reflect the distribution of the optimal solutions. The solution with a larger crowding distance has good performance in diversity. Table 1 is the pseudocode of the proposed non-dominated sorting multi-objective whale

optimization algorithm for detecting observable communities. A directed weighted graph is first constructed based on a state-space process model. The algorithm starts by initializing the population described in Ref [46]. It then maximizes the modularity index subject to the structural observability of each subsystem as described in Eq.(20). To this end, the position of each whale is compared with the positions of the other whales to rank those non-dominated solutions (positions) in an archive. The crowding distance is used to keep diversity along the Pareto optimal members. In every iteration, the position of each whale is updated using WOA equations. Solutions are calculated based on the updated positions, and the archive is updated accordingly. When the archive is updated, the best solution (leader) is chosen by random from the first Pareto front. This operation continues until the maximum number of iterations is reached. Each of these generated solutions corresponds to the observable communities with different cluster numbers.

3.3. Parallel Computation

In the design of state estimators, performance (rate of convergence to true values) and robustness (low sensitivity of the estimates to system model uncertainty and unmeasured input disturbances) are of importance. There is a tradeoff between these two properties that should be considered in practice. Reduced-order estimators have been proposed for large-scale models[49]. Distributed state estimation architectures that run in parallel, offer several advantages over their centralized counterparts, including scalability, flexibility, and robustness[24]. Furthermore, parallel computing allows for reducing the computational cost of the implementation for large-scale processes. This study develops a parallelization scheme consisting of the following steps: (i) decomposing the entire problem into small problems; (ii) assigning tasks to existing processors; and (iii) communicating among the involved processors to exchange information. Such communication leads to the exchange and synchronization of data and tasks between all of the processors.

The proposed parallelization scheme utilizes a master-slave (MS) scheme. A master processor is responsible for decomposing the entire problem into small problems and allocating tasks to other processors (slave processors). The salve processors are in charge of carrying out their assigned tasks. Each slave processor focuses on completing its assigned tasks and updating the master processor about the results. Before initiating their next task executions, all processors wait until they have received all the data computed by the other processors at the previous task

execution step. It is worth noting that parallelization lowers computational costs when the optimization problem is sufficiently large. In order words, the computational time needed to solve small problems increases with parallelization, as the time needed to execute individual tasks on different processors is comparable to the time needed to communicate information between the processors. In the proposed algorithm, we use the parallelization scheme in two separate sections: a) in the community detection framework and b) in the distributed state estimator scheme. In the community detection framework, the master processor is in charge of assigning the global search operations, while slave processors do the objective function evaluations. In parallel computation applied to the distributed state estimation, the local state estimation tasks are executed by slave processors while the master processor exchanges information for each time instant between the

local estimators. Table 2 summarizes the parallel computations in the proposed algorithm for implementing the distributed state estimators.

Table 1: Pseudocode of proposed algorithm for decomposing a large-scale process into observable subsystems.

Input: Adjacency matrix of the process

Output: Observable subsystems in the first Pareto front

Begin:

Set iteration number $t_{Current} = 0$;

Generate the ordered neighbor list

Initialize N populations based on modified LAR described in Ref [50]

Compute the objective functions subject to described constraints according to Eq. (20) and choose the leader

Store the observable nondominated solutions into an archive

Sort solutions according to non-domination rank as described in Ref [48]

Compute the crowding distance for each non-dominated solution stored in the archive

While $t_{Current} < t_{Max\ Iter}$,

For each whale,

Do the whale optimization algorithm to update the position of each whale as specified in Ref [50]

End for

Calculate the objective functions and check the structural observability index of the solutions, O

Sort solutions according to non-domination rank

Compute the crowding distance for each non-dominated solution stored in the archive

Update archive

Update leader by random from Pareto optimal Front

$t_{Current} = t_{Current} + 1$

End while

Return to archive

Until the maximum number of iterations is reached

4. Case Study

This section focuses on the implementation and validation of the proposed methods in two case studies.

4.1. Reactor-separator process

This reactor-separator process consists of two continuous stirred tank reactors (CSTRs) and one flash tank separator, as shown in Figure 3 [1,50]. The exothermic first-order series reactions $A \xrightarrow{r_1} B \xrightarrow{r_2} C$ take place inside the reactors before the outlet stream from the second CSTR

is fed to the flash tank. The overhead gas stream from the separator is condensed and is partially recycled to the first CSTR. The bottom stream of the separator is the product stream.

A mathematical model of the process is described in Ref.[1,50]. It is based on the following assumptions: well-mixed conditions; the physical properties such as density and heat capacity are constant; the two feed streams (F_{10}, F_{20}) contain only component A; the split ratio for each of the components remains constant within the operating temperature range of the flash tank.

$$\frac{dx_{A,1}}{dt} = \frac{F_{10}}{V_1}\left(x_{A,10} - x_{A,1}\right) + \frac{F_r}{V_1}\left(x_{A,r} - x_{A,1}\right) - k_1 e^{\frac{-E_1}{R\,T_1}} x_{A,1}$$

$$\frac{dx_{B,1}}{dt} = \frac{F_{10}}{V_1}\left(x_{B,10} - x_{B,1}\right) + \frac{F_r}{V_1}\left(x_{B,r} - x_{B,1}\right) + k_1 e^{\frac{-E_1}{R\,T_1}} x_{A,1} - k_2 e^{\frac{-E_2}{R\,T_1}} x_{B,1}$$

$$\frac{dT_1}{dt} = \frac{F_{10}}{V_1}(T_{10} - T_1) + \frac{F_r}{V_1}(T_3 - T_1) + \frac{-\Delta H_1}{C_p} k_1 e^{\frac{-E_1}{R\,T_1}} x_{A,1} + \frac{-\Delta H_2}{C_p} k_2 e^{\frac{-E_2}{R\,T_1}} x_{B,1} + \frac{Q_1}{\rho C_p V_1}$$

$$\frac{dx_{A,2}}{dt} = \frac{F_1}{V_2}\left(x_{A,1} - x_{A,2}\right) + \frac{F_{20}}{V_2}\left(x_{A,20} - x_{A,2}\right) - k_1 e^{\frac{-E_1}{R\,T_2}} x_{A,2}$$

$$\frac{dx_{B,2}}{dt} = \frac{F_1}{V_2}\left(x_{B,1} - x_{B,2}\right) + \frac{F_{20}}{V_2}\left(x_{B,20} - x_{B,2}\right) + k_1 e^{\frac{-E_1}{R\,T_2}} x_{A,2} - k_2 e^{\frac{-E_2}{R\,T_2}} x_{B,2}$$

$$\frac{dT_2}{dt} = \frac{F_1}{V_2}(T_1 - T_2) + \frac{F_{20}}{V_2}(T_{20} - T_2) + \frac{-\Delta H_1}{C_p} k_1 e^{\frac{-E_1}{R\,T_2}} x_{A,2} + \frac{-\Delta H_2}{C_p} k_2 e^{\frac{-E_2}{R\,T_2}} x_{B,2} + \frac{Q_2}{\rho C_p V_2}$$

$$\frac{dx_{A,3}}{dt} = \frac{F_2}{V_3}\left(x_{A,2} - x_{A,3}\right) - \frac{F_r + F_p}{V_3}\left(x_{A,r} - x_{A,3}\right)$$

$$\frac{dx_{B,3}}{dt} = \frac{F_2}{V_3}\left(x_{B,2} - x_{B,3}\right) - \frac{F_r + F_p}{V_3}\left(x_{B,r} - x_{B,3}\right)$$

$$\frac{dT_3}{dt} = \frac{F_2}{V_3}(T_2 - T_3) + \frac{Q_3}{\rho C_p V_3} \qquad (21)$$

where $x_{A,j}$ and $x_{B,j}$ describe the mass fractions of A and B in the jth vessel, respectively, and T_j is the temperature inside the jth vessel. The model parameter values are given in Table 3. We assume that only temperature measurements are available. The proposed method is applied to decompose

Table 2: Parallel implementation of the proposed algorithm.

Master Processor		
Send data to slave processors Receive data from all slave processors Exchange the results	**Community Detection** *t*=0 Initialize *N* populations (*N* is the number of whales) Send *N/S* populations to each slave processor (*S* is the number of processors) While (**Not** stopping criteria [PS 1]) Receive computed fitness values from slave processors, choose the leader, and update all of the whale positions to obtain new populations, update best solutions in the archive	**State Estimation** *t*=0 For *i*=1 to *l* (*l* is the number of subsystems), do the following activities: Set the parameters and initial guesses Assign estimation task to slave processors Receive the executed estimation tasks from each slave processor and share them with other slave processors
Slave Processors		
Receive data Compute assigned tasks Send the results to the master processor	*t*=0 Receive *N/S* populations Evaluate *N/S* populations Send the evaluated fitness function to the master processor While (**Not** stopping criteria [PS 1]) Evaluate N/S populations based on the updated archive	Receive the local measurements Execute estimation based on local measurements and data received from the master processor about the last time instant of other slave processors Send the results to the master processor Wait till other processors finish their tasks and share their information with the master
PS 1: The maximum allowable iteration number is considered stopping criteria to avoid an increase the in computation time.		

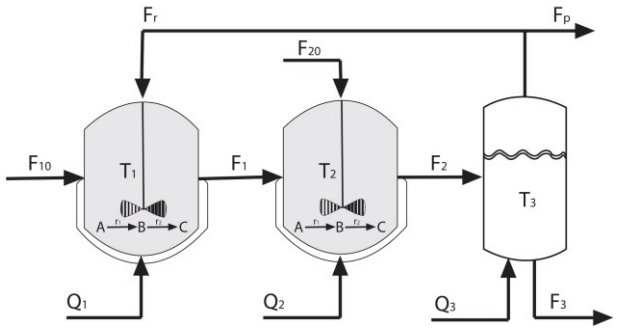

Figure 3: Reactor–separator process with recycle.

the reactor-separator process into observable subsystems. To this end, the weighted adjacency matrix showing interconnections between each pair of variables is constructed based on the state-space model of Eq. (21). The strength of the interconnections between each variable pair are determined using sensitivity analyses. The structural observability matrix given in Definition.1 is formed accordingly. A whale population size of 50 is used, and the maximum iteration number as a stopping criterion is set to 250. The results indicate the maximum modularity that decomposes the process into observable subsystems is about 0.451, which corresponds to three subsystems. The assigned state and measured variables to each subsystem are reported in Table 4. In addition to the observability, this decomposition captures the physical topology of the plant, which may result in reducing the computational complexity.

Table 3: Reactor–separator process parameters.

Variables	Description	Value (Unit)
$\boldsymbol{F_{10}, F_{20}}$	Feed flow rates to vessels 1,2	5.04 (m^3/hr)
$\boldsymbol{V_i}$	Volumes of vessel $j,\ j = 1,2,3,$ respectively	1, 0.5, 1 (m^3)
$\boldsymbol{F_r}$	Flow rate of the recycle	17 (m^3/hr)
$\boldsymbol{F_p}$	Flow rate of the purge	0.34 (m^3/hr)
$\boldsymbol{k_1}$	Pre-exponential values for reactions 1	9.97×10^6 (hr^{-1})
$\boldsymbol{E_1}$	Activation energy for reactions 1	50 (kJ/mol)
$\boldsymbol{k_2}$	Pre-exponential values for reactions 1	9×10^6 (hr^{-1})
$\boldsymbol{E_2}$	Activation energy for reactions 2	60 (kJ/mol)
$\boldsymbol{T_{10}, T_{20}}$	Temperatures of inlet streams of vessels 1 and 2	359 (°K)
$\boldsymbol{\Delta H_1}$	Heat of reaction for reactions 1	-60 (kJ/mol)
$\boldsymbol{\Delta H_2}$	Heat of reaction for reactions 2	-70 (kJ/mol)
$\boldsymbol{Q_i}$	Heat input to the vessel $j,\ j = 1,2,3$	715.3, 579.8, 568.7 (MJ/hr)
$\boldsymbol{C_p}$	Heat capacity	4.2 (KJ/kg. °K)
$\boldsymbol{R}$	Gas constant	8.314 (J/°K mol)
$\boldsymbol{\rho}$	Solution density	1000 (kg/m^3)
$\boldsymbol{x_{A,10}, x_{B,10}}$	Mass fractions of A and B in inlet stream to vessel 1	1,0
$\boldsymbol{x_{A,20}, x_{B,20}}$	Mass fractions of A and B in inlet stream to vessel 2	1,0
$\boldsymbol{\alpha_A, \alpha_B, \alpha_C}$	Split Ratio of A, B, and C	5, 1, 0.5

Table 4: Variables of the reactor-separator process found observable subsystems.

Subsystem	State variables	Measured variables
1	$x_{A,1}, x_{B,1}, T_1$	T_1
2	$x_{A,2}, x_{B,2}, T_2$	T_2
3	$x_{A,3}, x_{B,3}, T_3$	T_3

Considering this decomposition, the distributed state estimation algorithms described in sections 2.1 and 2.3 are implemented for the reactor-separator process. Each of the filters starts with the initial state estimate $\hat{x}_0^+ = 1.05\, x_0$ and the initial estimation-error covariance $P_0^+ = Q$, where $Q = \text{diag}(10^{-2} \times \|x_0\|^2)$, and $R = \text{diag}(10^{-4} \times \|y_0\|^2)$. For the distributed estimators, the initial state estimates are the same as those for the centralized estimators. Figure 4 compares the actual values and estimates of the state variables obtained using the distributed estimators. As can be seen, all extensions can track the trajectory of the actual system well. Moreover, all estimators show similar performance at steady-state conditions, while there is some difference in the performance of filters before the variables reach the steady-state condition, as shown in Figure 5. Among these distributed estimators, SUKF converges faster to the noise covariance and thus achieves a better performance in terms of robustness, accuracy, and computation for nonlinear estimation, which is demonstrated by the simulation results. These results motivate us to investigate the performance of SUKF in the centralized and distributed schemes in detail. We have also implemented our proposed parallelization scheme to explore how this scheme reduces the running time.

Before we proceed, first, the effect of random noise on the measurements and process variables is considered. Since noise vectors are randomly generated on the basis of the known PDF of w_{k-1} and v_k, it is difficult to verify the results of different estimators in a single run. To avoid this, we apply the Monte Carlo simulation method using repeated sampling to obtain results by assigning random values to the uncertain variables. Once the simulation is complete for all of the simulations, the results are averaged to provide estimates.

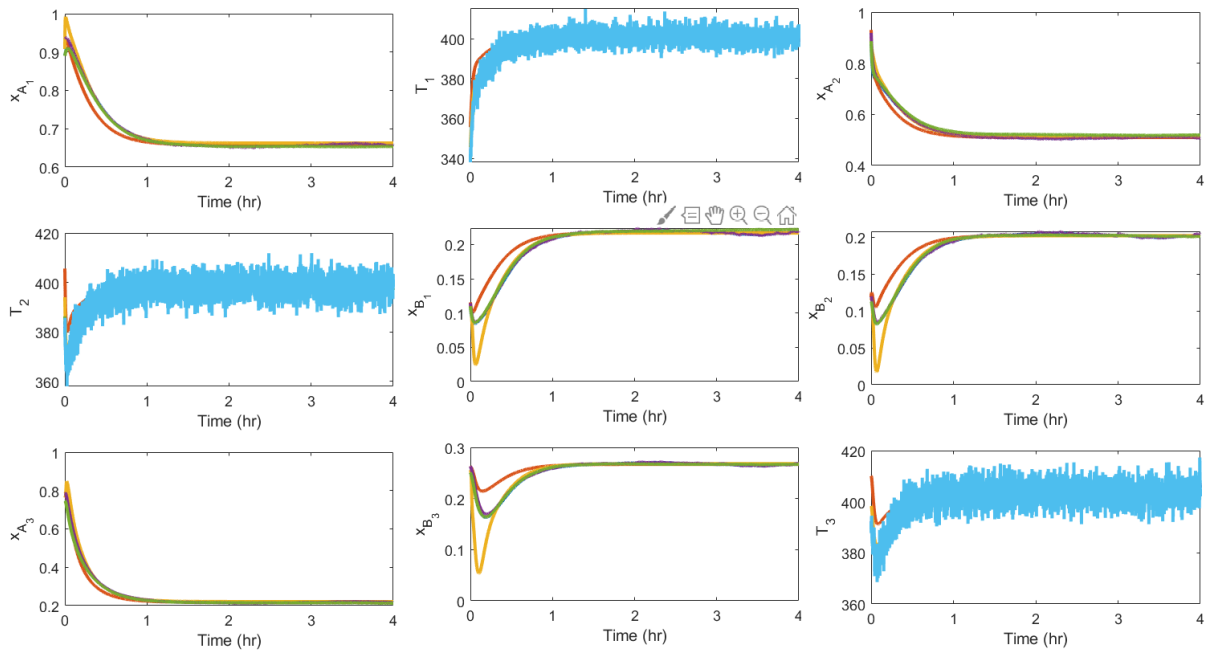

Figure 4: True values (blue lines) and measurements (light blue lines) of the state variables. Estimates of the state variables calculated by the distributed EKF (red lines), the distributed CKF (yellow lines), the distributed UKF (purple lines), and the distributed SUKF (green lines).

To compare the performance of the centralized and the distributed estimators, we compare the standard deviations of the estimation error by computing the standard deviations of the N (Monte Carlo simulation) estimation errors for each time step according to:

$$STD = \sqrt{\frac{1}{N-1}\sum_{i=1}^{i=N}(\hat{x}_i(t) - x_i(t))^2} \tag{22}$$

In this case study, we calculate and compare the standard deviations of the estimation errors for the distributed and centralized schemes of SUKF by running 50 Monte Carlo simulations, as shown in Figure 6. These results indicate no significant difference between the mean error values associated with those two schemes. Note that the distributed scheme may improve the mean error compared to the centralized scheme.

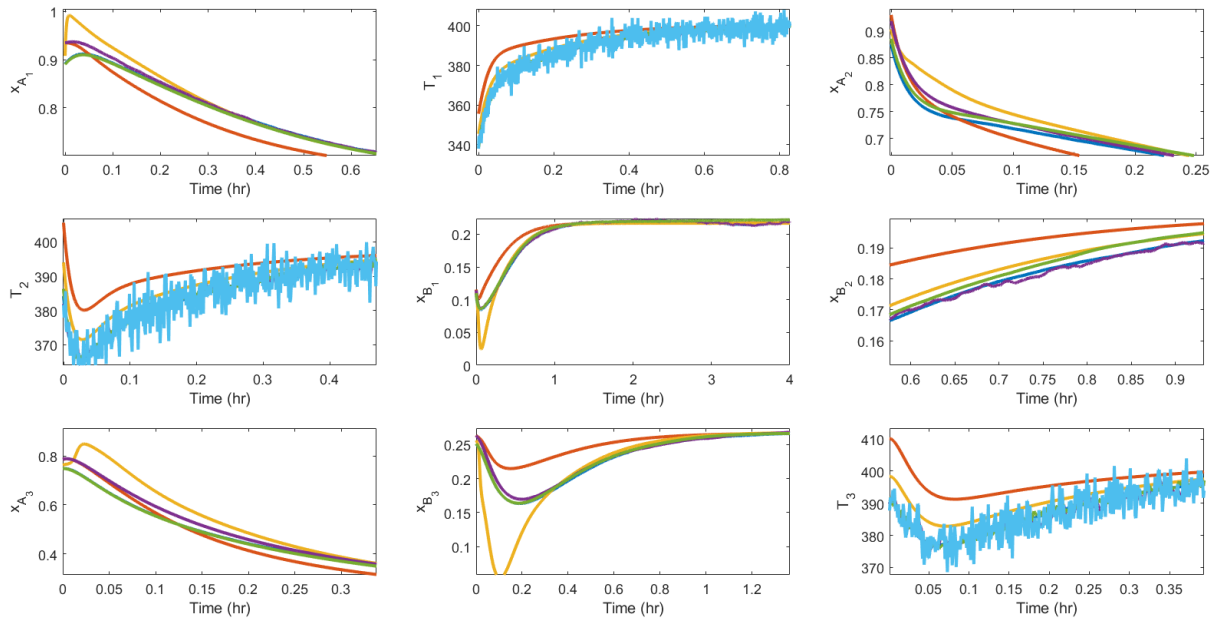

Figure 5: True values (blue lines) and measurements (light blue lines) of the state variables. Estimates of the state variables calculated by the distributed EKF (red lines), the distributed CKF (yellow lines), the distributed UKF (purple lines), and the distributed SUKF (green lines).

The parallel computation scheme introduced in section 3.4 is applied to the distributed framework. The hardware used in this work is Intel® Core™ i7 with a CPU 3.4 GHz and 32 GB of memory. Programming is done in MATLAB 2018b. Figure 7 indicates that the parallel implementation of the estimator requires about 20% less CPU time without sacrificing accuracy.

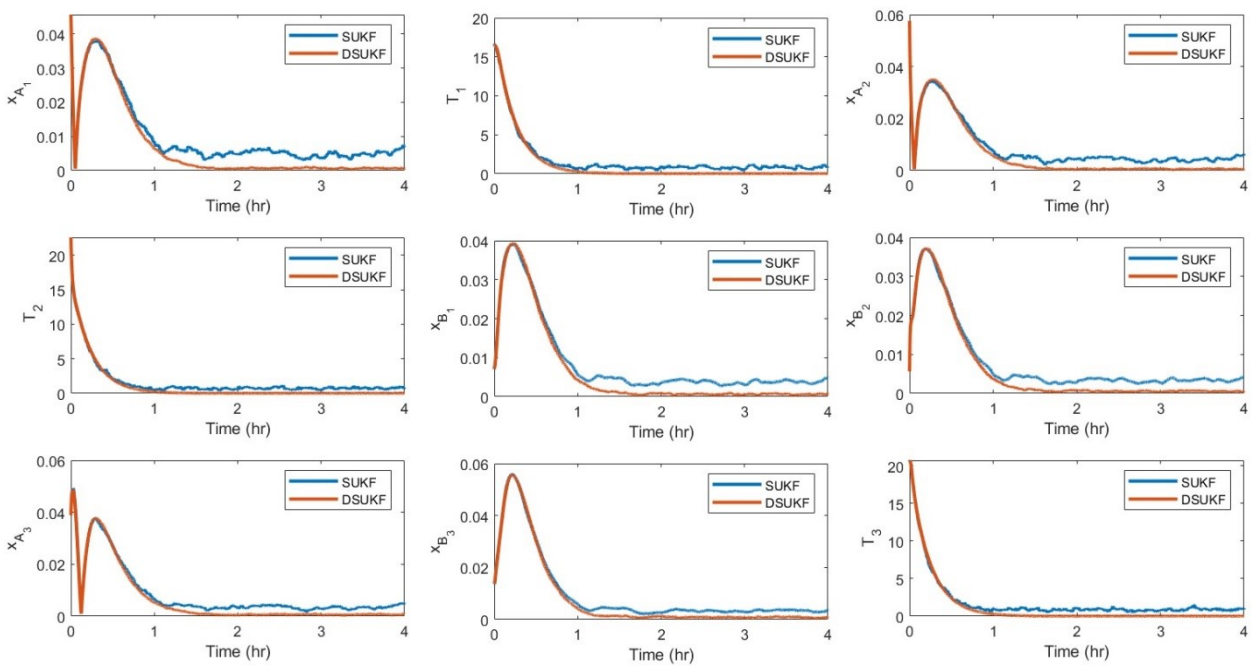

Figure 6: Standard deviations of the estimation errors. Distributed (red lines) and centralized (blue lines) SUKF.

4.2. Tennessee Eastman Process (TEP)

The proposed algorithm is tested on the Tennessee Eastman process (TEP)[51]. The TEP consists of five main components: a reactor, a condenser, a centrifugal compressor, a vapor-liquid separator, and a stripper. The mathematical model[51,52] is based on the following assumptions: all vapors behave as ideal gases; vapor/liquid equilibria follow Raoult's law with the vapor pressure calculated using the Antoine equation, and all vessels are assumed to be perfectly mixed.

$$\frac{dN_{i,r}}{dt} = y_{i,6}F_6 - y_{i,7}F_7 + \sum_{j=1}^{3} \nu_{ij}\, R_j \qquad i = A, B, \dots, H$$

$$\frac{dN_{i,s}}{dt} = y_{i,7}F_7 - y_{i,8}(F_8 + F_9) - x_{i,10}F_{10} \qquad i = A, B, \dots, H$$

$$\frac{dN_{i,m}}{dt} = z_{i,1}F_1 + z_{i,2}F_2 + z_{i,3}F_3 + y_{i,5}F_5 + y_{i,8}F_8 + F_i^* - y_{i,6}F_6 \qquad i = A, B, \dots, H$$

$$\frac{dN_{i,p}}{dt} = (1 - \varphi_i)x_{i,10}F_{10} - x_{i,11}F_{11} \qquad i = G, H \tag{23}$$

The measurements are assumed to be available at each sampling instant. Moreover, all measurements come in without delay. All control variables are kept at their base case level as reported in Ref[51,52]. The proposed decomposition method is applied to decompose the TEP into observable subsystems. To do that, the adjacency matrix of the process is constructed based on the state-space model in Eq.(23). The strength of the interconnections between each variable pair is determined using sensitivity analyses. The elements of the adjacency matrix are weighted accordingly. The results allow determining the 58×58 weighted adjacency matrix. The structural observability matrix given in Definition1 is formed. A whale population size of 50 is used, and the maximum iteration number as a stopping criterion is set to 250. The results show that the TEP is decomposed into three subsystems in which all communities are structurally observable for the distributed implementation of state estimators. The reactor and feeding zone are in the first subsystem, the separator in the second subsystem, and the stripper in the third subsystem. The modularity corresponding to these communities is about 0.374.

According to the dynamic model of the TEP, there are 26 state variables, including the molar holdup of all the reactants in the mixing zone, reactor, separator, and stripper. The direct measurement of these state variables is difficult, but their measurement plays an important role in monitoring, control, and process model accuracy. Hence, the estimation of these states is justified for the reasons mentioned above. A total of 30 process variables are measured in the process according to Ref[51,52].

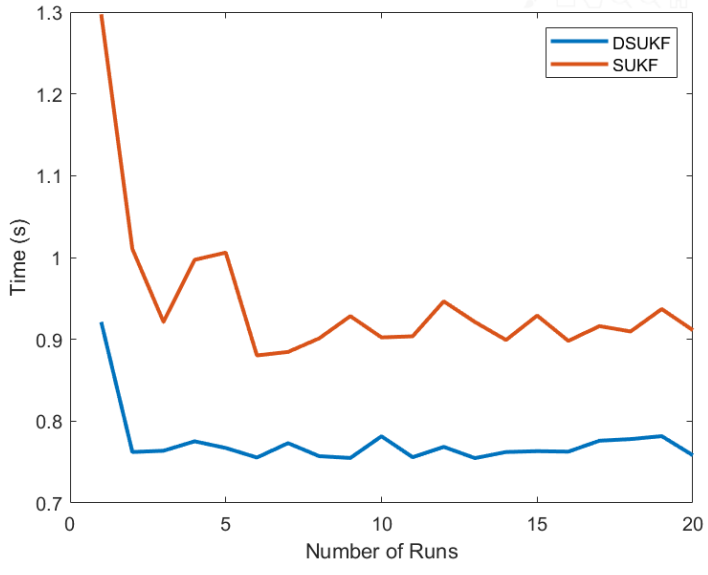

Figure 7: CPU time vs. the number of runs.

Since there are many measured outputs and state variables in the TEP, showing all state variable estimates are not possible. Here, only some of the state estimates are shown. Different state estimation methods have been implemented centrally to estimate the state variables of the Tennessee Eastman Process [53-55]. In particular, the central implementations of EKF and UKF state estimators were tested on the TEP[55, 57,] and the performances of the estimators were compared. A moving horizon estimator was also applied to the TEP by fusing past measurements within a given time horizon and calculating state estimates based on the maximum-likelihood principle[56]. To evaluate the performances of the four estimators, we assume that $\hat{x}_0^+ = 1.05\, x_0$, $Q = \text{diag}(10^{-2} \times x_0)$, and $R = \text{diag}(10^{-1} \times y_0)$. The plots in the top row of Figure 8 compare the measured, estimated, and true values of the volume of the liquid in the reactor (V_{lr}) and the molar concentrations of the products G and H. They show that despite the large measurement noise signals, the four estimators provide adequately accurate state estimates. The plots in the bottom row of Figure 8 show the true values of the rate of the production of G in the reactor ($R_{G,r}$), the molar holdup of G in the reactor ($N_{G,r}$), and molar holdup of G in the separator ($N_{G,s}$) and their estimated values calculated using the four distributed state estimators. The results demonstrate the satisfactory performances of distributed UKF and SUKF estimators.

To compare the performances of the centralized and distributed estimators, we calculate the estimation errors by conducting 50 Monte Carlo simulations. Table 5 compares the root mean square errors (RMSEs) of the four distributed estimators for some state variables. The results indicate the poor performance of the distributed EKF due to the poor accuracy of the linear approximation. However, as the distributed UKF and SUKF do not use the linear approximation, their performances are much better.

Table 5. RMSE of the distributed EKF, UKF, and SUKF for some variables.

State Variables	**DEKF**	**DCKF**	**DUKF**	**DSUKF**
Component G in Product	0.541	0.321	**0.267**	0.308
Component H in Product	1.167	0.484	0.431	**0.309**
Molar Holdup G in the Reactor	1.881	0.961	0.682	**0.442**
Molar Holdup G in the Separator	3.451	0.923	**0.321**	0.371

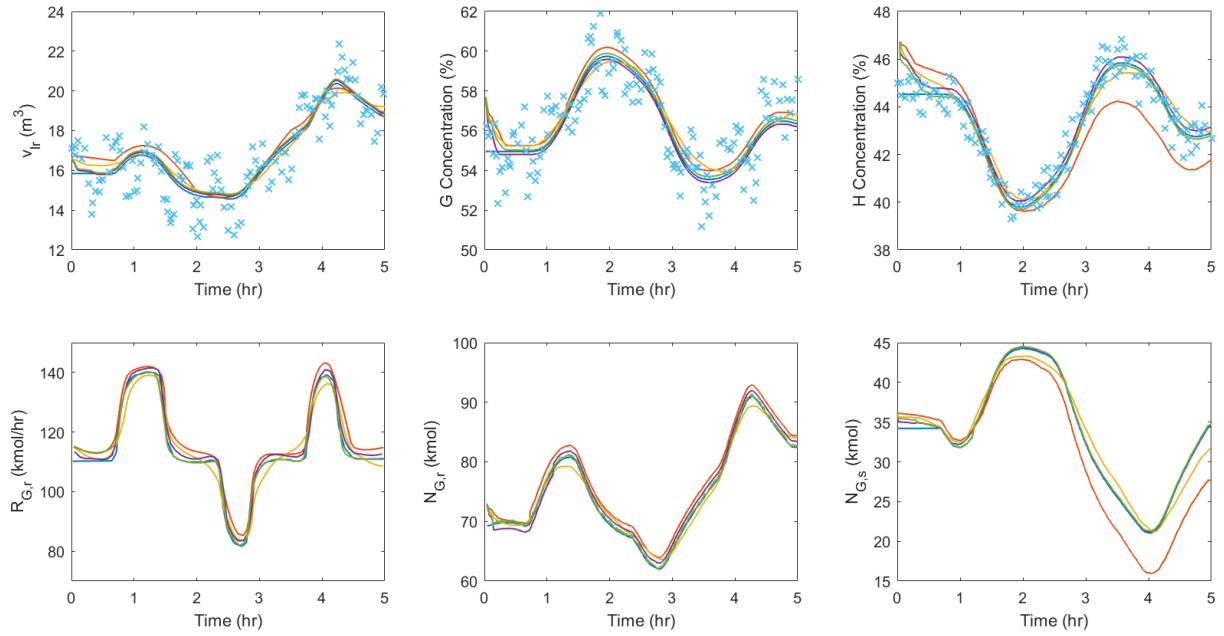

Figure 8: True values (blue lines) and measurements (cross points) of the state variables. Estimates of the state variables calculated by the distributed EKF (red lines), the distributed CKF (yellow lines), the distributed UKF (purple lines), and the distributed SUKF (green lines).

We also compare the performances of the parallel-implemented distributed UKS and SUKF with those of their centralized counterparts to see how parallelization reduces the running time. This comparison (Figure 9) indicates that the centralized estimators require 20% longer CPU time than their distributed counterparts. Furthermore, the SUKF requires fewer calculations than the UKF, as it utilizes fewer sigma points to approximate the states' probability distribution.

5. Conclusion

We studied distributed implementation of state estimators in large-scale systems. To this end, we proposed an algorithm that decomposes a large-scale system into structurally observable subsystems. The proposed algorithm is based on community detection in a directed weighted graph constructed from a state-space process model. We solved the community detection problem as a multi-objective optimization problem using the Whale optimization algorithm. To make sure that each subsystem is observable from its local measurements, we proposed a systematic approach for

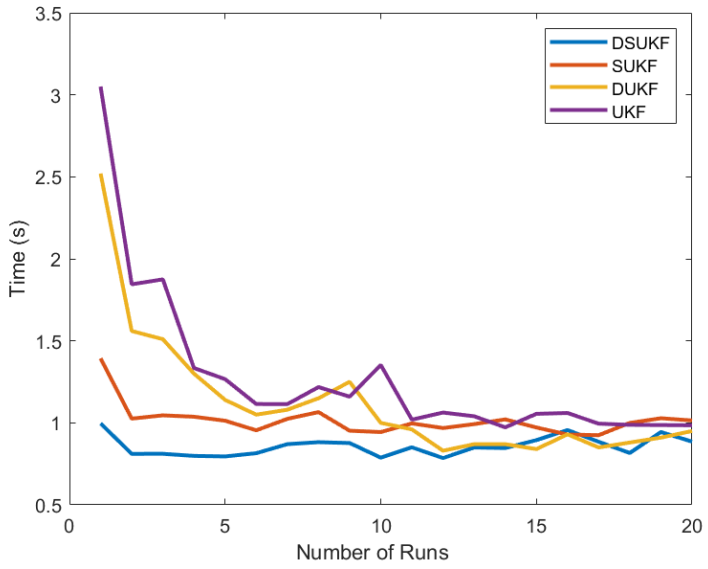

Figure 9: CPU time vs. the number of runs

checking the structural observability of a system. This approach used the degrees of interactions among variables measured by sensitivity analysis. The proposed algorithm is implemented and validated in two case studies. In each case study, the distributed Kalman filter extensions are implemented in the subsystems derived from the proposed decomposition algorithm to compare their performance with their centralized counterparts. We also proposed a parallelization strategy that achieves superior performance without compromising accuracy. The parallelization scheme is used a) in solving multi-objective optimization problems to identify communities and b) in executing the local state estimation at every sampled time instant. The results showed that the distributed configuration reduced the computational burden in local estimations, and the parallelization scheme improved the computational efficiency.

An application of this work is in the design of partial state observers/estimators, which estimate a subset of the state variables that are of importance. Partial state observers/estimators are much easier to design and are more robust. In this case, the community detection problem is formulated as that of finding the subset of input and output measurements that are essential to reliably estimate the important state variables. The solution to this problem will be the subset of measurable inputs that strongly affect the important state variables and the subset of measurable outputs from which the important state variables are observable. A partial state observers/estimator

is then designed to estimate the important state variables from these essential input and output measurements.

Acknowledgment

This material is based upon work supported by the U.S. National Science Foundation under Grant Nos. CBET–1704915 and CBET–1704833. Any opinions, findings, and conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Christofides PD, Liu J, De La Pena DM. *Networked and distributed predictive control: Methods and nonlinear process network applications.* Springer Science & Business Media; 2011.
2. Soroush M, Masooleh LS, Seider WD, Oktem U, Arbogast JE. Model-predictive safety optimal actions to detect and handle process operation hazards. *AIChE Journal.* 2020;66(6):e16932.
3. Soroush M, Masooleh LS, Arbogast JE, Seider WD, Oktem U. Model-predictive safety: A new evolution in functional safety. In: *Smart Manufacturing.* Elsevier; 2020:283-321.
4. Stavropoulos P, Chantzis D, Doukas C, Papacharalampopoulos A, Chryssolouris G. Monitoring and control of manufacturing processes: A review. *Procedia CIRP.* 2013;8:421-425.
5. Spivey BJ, Hedengren JD, Edgar TF. Constrained nonlinear estimation for industrial process fouling. *Industrial & Engineering Chemistry Research.* 2010;49(17):7824-7831.
6. Soroush M. Nonlinear state-observer design with application to reactors. *Chemical Engineering Science.* 1997;52(3):387-404.
7. Soroush M. State and parameter estimations and their applications in process control. *Computers & Chemical Engineering.* 1998;23(2):229-245.
8. Kalman RE. A new approach to linear filtering and prediction problems. 1960.
9. Luenberger DG. Observing the state of a linear system. *IEEE transactions on military electronics.* 1964;8(2):74-80.
10. Kazantzis N, Kravaris C. Nonlinear observer design using Lyapunov's auxiliary theorem. *Systems & Control Letters.* 1998;34(5):241-247.
11. Valluri S, Soroush M. Nonlinear state estimation in the presence of multiple steady states. *Industrial & engineering chemistry research.* 1996;35(8):2645-2659.
12. Dochain D. State and parameter estimation in chemical and biochemical processes: a tutorial. *Journal of process control.* 2003;13(8):801-818.
13. Julier SJ, Uhlmann JK. New extension of the Kalman filter to nonlinear systems. Paper presented at: Signal processing, sensor fusion, and target recognition VI1997.

14. Julier S, Uhlmann J, Durrant-Whyte HF. A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on automatic control.* 2000;45(3):477-482.
15. Arasaratnam I, Haykin S. Cubature kalman filters. *IEEE Transactions on automatic control.* 2009;54(6):1254-1269.
16. Gordon NJ, Salmond DJ, Smith AF. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. Paper presented at: IEE proceedings F (radar and signal processing)1993.
17. Frank PM. Fault diagnosis in dynamic systems via state estimation-a survey. In: *System fault diagnostics, reliability and related knowledge-based approaches.* Springer; 1987:35-98.
18. Lima FV, Rawlings JB. Nonlinear stochastic modeling to improve state estimation in process monitoring and control. *AIChE journal.* 2011;57(4):996-1007.
19. Campani G, Ribeiro MP, Zangirolami TC, Lima FV. A hierarchical state estimation and control framework for monitoring and dissolved oxygen regulation in bioprocesses. *Bioprocess and biosystems engineering.* 2019;42(9):1467-1481.
20. Ali JM, Hoang NH, Hussain MA, Dochain D. Review and classification of recent observers applied in chemical process systems. *Computers & Chemical Engineering.* 2015;76:27-41.
21. Afshari HH, Gadsden SA, Habibi S. Gaussian filters for parameter and state estimation: A general review of theory and recent trends. *Signal Processing.* 2017;135:218-238.
22. Alexander R, Campani G, Dinh S, Lima FV. Challenges and opportunities on nonlinear state estimation of chemical and biochemical processes. *Processes.* 2020;8(11):1462.
23. Rawlings JB, Ji L. Optimization-based state estimation: Current status and some new results. *Journal of Process Control.* 2012;22(8):1439-1444.
24. Khan UA, Moura JM. Distributing the Kalman filter for large-scale systems. *IEEE Transactions on Signal Processing.* 2008;56(10):4919-4935.
25. Carli R, Chiuso A, Schenato L, Zampieri S. Distributed Kalman filtering based on consensus strategies. *IEEE Journal on Selected Areas in communications.* 2008;26(4):622-633.
26. Pourkargar DB, Almansoori A, Daoutidis P. Impact of decomposition on distributed model predictive control: A process network case study. *Industrial & Engineering Chemistry Research.* 2017;56(34):9606-9616.
27. Tang W, Daoutidis P. Network decomposition for distributed control through community detection in input–output bipartite graphs. *Journal of Process Control.* 2018;64:7-14.
28. Newman ME, Girvan M. Finding and evaluating community structure in networks. *Physical review E.* 2004;69(2):026113.
29. Pourkargar DB, Moharir M, Almansoori A, Daoutidis P. Distributed estimation and nonlinear model predictive control using community detection. *Industrial & Engineering Chemistry Research.* 2019;58(30):13495-13507.
30. Yin X, Liu J. Distributed state estimation for a class of nonlinear processes based on high-gain observers. *Chemical Engineering Research and Design.* 2020;160:20-30.

31. Zhang L, Yin X, Liu J. Complex system decomposition for distributed state estimation based on weighted graph. *Chemical Engineering Research and Design.* 2019;151:10-22.
32. Yin X, Arulmaran K, Liu J, Zeng J. Subsystem decomposition and configuration for distributed state estimation. *AIChE journal.* 2016;62(6):1995-2003.
33. Lima FV, Rajamani MR, Soderstrom TA, Rawlings JB. Covariance and state estimation of weakly observable systems: Application to polymerization processes. *IEEE Transactions on Control Systems Technology.* 2012;21(4):1249-1257.
34. Simon D. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches.* John Wiley & Sons; 2006.
35. Bell BM, Cathey FW. The iterated Kalman filter update as a Gauss-Newton method. *IEEE Transactions on Automatic Control.* 1993;38(2):294-297.
36. Särkkä S. *Bayesian filtering and smoothing.* Vol 3: Cambridge University Press; 2013.
37. Julier SJ, Uhlmann JK. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE.* 2004;92(3):401-422.
38. Julier SJ. The spherical simplex unscented transformation. Paper presented at: Proceedings of the 2003 American Control Conference, 2003.2003.
39. Zeng J, Liu J, Zou T, Yuan D. Distributed extended Kalman filtering for wastewater treatment processes. *Industrial & Engineering Chemistry Research.* 2016;55(28):7720-7729.
40. Hu Z, Hu J, Yang G. A survey on distributed filtering, estimation and fusion for nonlinear systems with communication constraints: new advances and prospects. *Systems Science & Control Engineering.* 2020;8(1):189-205.
41. Cattivelli FS, Sayed AH. Diffusion strategies for distributed Kalman filtering and smoothing. *IEEE Transactions on automatic control.* 2010;55(9):2069-2084.
42. Hlinka O, Hlawatsch F, Djuric PM. Distributed particle filtering in agent networks: A survey, classification, and comparison. *IEEE Signal Processing Magazine.* 2012;30(1):61-81.
43. Farina M, Ferrari-Trecate G, Scattolini R. Distributed moving horizon estimation for nonlinear constrained systems. *International Journal of Robust and Nonlinear Control.* 2012;22(2):123-143.
44. Brandes U, Delling D, Gaertler M, et al. On modularity clustering. *IEEE transactions on knowledge and data engineering.* 2007;20(2):172-188.
45. Saka MP, Hasançebi O, Geem ZW. Metaheuristics in structural optimization and discussions on harmony search algorithm. *Swarm and Evolutionary Computation.* 2016;28:88-97.
46. Masooleh LS, Arbogast JE, Seider WD, Oktem U, Soroush M. An Efficient Algorithm for Community Detection in Complex Weighted Networks. *AIChE Journal.* 2020.
47. Mirjalili S, Lewis A. The whale optimization algorithm. *Advances in engineering software.* 2016;95:51-67.
48. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation.* 2002;6(2):182-197.
49. Hernandez EM. A natural observer for optimal state estimation in second order linear structural systems. *Mechanical Systems and Signal Processing.* 2011;25(8):2938-2947.

50. Liu J, Munoz de la Pena D, Christofides PD. Distributed model predictive control of nonlinear process systems. *AIChE journal.* 2009;55(5):1171-1184.
51. Downs JJ, Vogel EF. A plant-wide industrial process control problem. *Computers & chemical engineering.* 1993;17(3):245-255.
52. Ricker N, Lee J. Nonlinear modeling and state estimation for the Tennessee Eastman challenge process. *Computers & chemical engineering.* 1995;19(9):983-1005.
53. Upendra JV, Prakash J. Comparison of State Estimation Algorithms on the Tennessee Eastman Process. In: *Recent Advancements in System Modelling Applications.* Springer; 2013:357-368.
54. Kraus T, Kuhl P, Wirsching L, Bock HG, Diehl M. A moving horizon state estimation algorithm applied to the tennessee eastman benchmark process. Paper presented at: 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems2006.
55. Kottakki KK, Bhushan M, Bhartiya S. Unconstrained Nonlinear State Estimation for Tennessee Eastman Challenge Process. *IFAC-PapersOnLine.* 2017;50(1):12919-12924.