A Privacy-preserving State Estimation Scheme for Smart Grids

Hong-Yen Tran, Jiankun Hu*, *Senior Member, IEEE,* and Hemanshu R. Pota

Abstract—With the appearance of electric energy market deregulation, there exists a growing concern over the potential privacy leakage of commercial data among competing power companies where data sharing is essential in the applications such as smart grid state estimation. Most of the existing solutions are either perturbation-based or conventional cryptography-based where a trusted central 3rd party would often be required. This paper proposes privacy-preserving state estimation protocols for DC and AC models. The proposed idea is to distribute the overall task of the system state estimation into sub-tasks which can be performed by local sub-grid operators with their private data. A masking method is designed inside a homomorphic encryption scheme which is then used to ensure both the input and output data privacy during the collaboration process among individual sub-task players. Security is achieved via the computationally indistinguishable post-quantum security guaranteed by a levelled homomorphic encryption scheme over real numbers and the differential privacy of the output estimated states provided by the Laplace mechanism perturbation integrated into the masking linear transformation. Simulation results are presented to demonstrate the validity of our proposed privacy-preserving system state estimation protocols.

Index Terms—smart grids, privacy-preserving, competitive privacy, power industry deregulation, state estimation, homomorphic encryption, perturbation.

1 Introduction

The prominent deployment of Advanced Metering Infrastructure (AMI), Phasor Measurement Units (PMUs) and other types of sensors and smart devices on power systems provides rich sources of energy data for various types of analytics, ranging from energy management to security operations on smart grids [1], [2]. One of the most critical operations in smart grids is system state estimation [3]. Smart grid system states can support many useful applications such as quick fault identification and outage restoration management, real time performance optimization, etc. [4]. Therefore, it is critical to obtain an accurate system state estimation for smart grid management.

Power industry deregulation is driving the need for state estimation of interconnected power systems [5]. In a deregulated power grid environment, independent transmission grid companies (TGCs) possess their internal measurements, line parameters, network topology, and states of the portion of the grid they manage. To estimate the system state of a power grid, certain information from all involved sub-grids must be shared [6]. However, competing power companies (TGCs) may be reluctant to contribute their private data due to the potential of compromising their interests [7] or the threats of cyber-physical attacks once the power grid data are accidentally or deliberately leaked to hackers (e.g, false data injection attacks [8], [9]). For example, the knowledge of the system states of neighboring power systems can be used to create competitive advantages on an electricity market by tuning a bidding strategy based on a good forecasting of location marginal price [5]. The more TGCs participate in a wider interconnected grid, the more complex and cumbersome regulatory and legal frameworks need to be established to govern the sharing of data between different operators. This also leads to the concern of competitive privacy between multiple power operators [10]. For convenience, the terms ''TGC' and ''sub-grid operator' will be used interchangeably. A grid model is represented as a set of interconnected buses attached with power generators and loads as in a standard power flow analysis.

State estimation in an interconnected power grid can be performed in an integrated (global) or distributed (local) manner. For an integrated estimator, the whole measurement set is collected and processed in a single state estimator. For a distributed approach [11], [12], each sub-grid performs local state estimation with local measurements taken within its area and then exchanges some information. The distributed approach is less accurate than the integrated one, but less information needs to be shared. Although the distributed estimator requires fewer data to be shared, the vulnerability of privacy leakage still exists. Thus, it is necessary to find solutions for privacy-preserving state estimation in interconnected transmission power systems that can achieve the integrated state estimation while protecting the privacy of individual sub-grids.

Under the constraint of controlling the trade-off between state estimation accuracy and the privacy of sub-grid operators, privacy-preserving state estimation in an interconnected transmission power grid requires non-trivial solutions. Encryption and perturbation are two primary

• *Hong-Yen Tran, Jiankun Hu* (corresponding author), and Hemanshu R. Pota are with the School of Engineering and Information Technology, University of New South Wales Canberra at ADFA, ACT 2602, Australia (e-mail: hongyen.tran@student.adfa.edu.au; j.hu@adfa.edu.au; h.pota@adfa.edu.au).*

Manuscript received ...; revised ...

tools for addressing privacy issues [13]. MPC [14] adopting a multi-key homomorphic encryption scheme which supports computation over encrypted data might be a candidate solution. The drawback is that a multi-key homomorphic encryption scheme is demanding in not only computation but also communication costs. The reason is that all parties have to generate partial decryption and share them so that it is able to decrypt or evaluate a function in the ciphertext space [15]. Masking the true data before sharing is another solution, namely perturbation. But it comes at the expense of state estimation accuracy degradation [16] or a requirement of a trusted centre and a secure distribution noise protocol [17], [18].

This paper examines the solutions to deal with the competitive privacy of transmission power companies when collaboratively implementing state estimation in an interconnected transmission power grid. From this perspective, privacy-enhanced versions for DC and AC state estimation are designed with the main contributions as follows:

- A novel idea is proposed that the overall task of privacy-preserving system state estimation, for both DC and AC models, is conducted by privacy-preserving collaborative computation over distributed pre-processed data from sub-grid operators.
- Privacy-preserving DC and AC state estimation protocols in a multi-area transmission grid are designed to provide a theoretical assurance of achieving state estimation accuracy and competitive privacy in a semi-honest adversarial model.
- Analysis of privacy regarding different types of adversaries is given. Privacy is mainly achieved via (1)- semantic security of a homomorphic encryption scheme and (2)- local differential privacy of the output estimated states with the Laplace mechanism integrated into a linear masking transformation.
- Experiments assessments are given and analyzed with the adoption of parallel matrix computation on high-performance computing to demonstrate the accuracy, efficiency, and scalability.

This paper consists of eight sections. Following this Introduction section are the Related Works and Preliminaries sections. The system model and threat model are presented in Section 4, which is followed by the description and analysis of the proposed protocols in Sections 5 and 6. Empirical evaluation is provided in Section 7. Finally, Section 8 is for the conclusions.

2 Related works

The trade-off between privacy and state estimation accuracy was investigated in an information-theoretic framework [7], [10]. The study in [16] proposed a privacy-preserving state estimation method in a single feeder of a distribution network based on load measurements from smart meters. Consumers' meter readings are perturbed with Laplace or Gaussian noise to achieve differential privacy. While the perturbation method provides differential privacy, it affects the quality of state estimation. In addressing this problem, a privacy-preserving state estimation scheme was proposed based on the perturbation of meter readings at the distribution level of a power grid in protecting consumers' privacy while still allowing a distribution operator to implement accurate state estimation [17]. The noise elements perturbed meter readings are centrally calculated to exploit the kernel of an electric grid configuration matrix supporting noise-cancelling in a state estimation process; thus, it does not affect the quality of state estimation as in [16]. The limitation is that the obfuscation necessitates a trusted lead smart meter to distribute each noise element to each designated meter. If this lead smart meter is compromised or acts as a semi-honest adversary, the scheme is insecure. The work in [18] improved [17] by splitting the process of generating and distributing the obfuscation vector among multiple gateways to reduce this vulnerability. However, this approach also does not provide a secure noise distribution protocol and still requires a trusted third party. Moreover, the issue of missing data (e.g., due to a communication loss) was not considered, whereas this problem could destroy the distortion error-free property, consequently affecting the correctness of state estimation. The system models of the state estimation schemes in [17], [18] are centralized with only one distribution system operator who carries out state estimation for a distribution grid; thus, only consumers' meter readings are required to be kept private. The configuration matrix (i.e, system parameters) and the estimated states are available for the distribution system operator in the scheme.

Considering a different privacy scenario in a transmission system, articles [10], [19], [20], [21] dealt with the threats of privacy leakage when state estimation is collaboratively processed between k sub-transmission systems. Existing approaches to decentralized state estimation partition the measurement vector such that each local area-based player attempts to perform local state estimation with measurements taken within its area. To estimate system-wide state variables, sub-systems need to share a portion of their data with others, but this raises the threat of breaching the private information of each sub-system. For a specific problem of distributed linear state estimation, [10] presented a trade-off between estimation fidelity and leakage of private state data as a result of sharing data in a two-agent network model. In [19], [20] a privacy-preserving distributed state estimation with phasor measurement units was investigated but the scheme still violates the privacy of sub-systems due to the exchange of information related to measurements on tie-lines linking neighbouring area. To securely contribute private information for a privacy-preserving hierarchical state estimation, [21] proposed a privacy-preserving distributed state estimation framework in which a cloud-based high-level control centre coordinates low-level control centres to compute the estimated states. The scheme requires multi-key homomorphic encryption to implement secure multi-party computation.

In this paper, we consider the privacy scenario in a multi-area transmission grid system. This is a competitive privacy problem [7] amongst the transmission system companies that have the conflict between the need of sharing data to estimate global system states with high accuracy (utility) and the need to withhold data (privacy) for competitive reasons. The proposed protocols focus on protecting the

private data of a transmission grid company, not only its input data (meter measurement readings, power line parameters, network topology) but also its output estimated states, while simultaneously achieving the accuracy of the integrated state estimation. The solution is a hybrid of an obfuscation technique guaranteeing differential privacy and a single-key post-quantum secure homomorphic encryption scheme established in a two non-colluding server model. The two-non-colluding server model [22] is a core architecture commonly used by previous works on privacy-preserving machine learning (e.g., see [23], [24], [25], [26]) where no server is trusted to handle the clear data. In this core setting, after collecting private data in protected forms (often in encryption) from many data-owners, the two servers then securely compute the model in a 2-party-secure-computation setting. The first server works on the encrypted data over the key of the second server while the second server works on the transformed data over the random elements of the first server. As long as the two servers do not collude, the security is guaranteed. This approach can help reduce the complexity of secure multi-party computation. A single-key homomorphic encryption scheme working on real numbers and achieving post-quantum security [27] is adopted. The proposed masking technique guarantees the differential privacy of the output estimated states.

3 PRELIMINARIES

3.1 Notations and definitions

Column vectors are denoted by lower-case bold letters, like $\boldsymbol{v}$. The i-th entry of the vector $\boldsymbol{v}$ is v_i. $\boldsymbol{v}^{\mathrm{T}}$ is the transpose of the column vector $\boldsymbol{v}$. Matrices are denoted by upper-case bold letters, like $\boldsymbol{A}$, where $\boldsymbol{A}^{\mathrm{T}}$ is the transpose of the matrix $\boldsymbol{A}$, and $\boldsymbol{A}^{-1}$ is its inverse. Either the zero-vector or the zero-matrix is represented by $\mathbf{0}$, which will be clear from the context. Given a set $\mathcal{S}$, $x \leftarrow_{\$} \mathcal{S}$ indicates that x is sampled uniformly at random from $\mathcal{S}$. The sup-norm of a vector is defined by $\|\boldsymbol{x}\|_\infty = \max_i\{|x_i|\}$. The notions used in this paper are given in Table 1.

TABLE 1: Notations used in the proposed privacy-preserving state estimation schemes

Notation	Description
N	Number of buses
n	Number of state variables
m	Number of devices (measurement readings)
k	Number of transmission grid companies (TGCs)
σ_i^2	Variance of reading errors of the i-th measuring device
$\mathcal{G}_i$	Sub-grid managed by TGC$_i$
$\mathcal{G}_0$	Interconnection grid managed by SO
n_i	Number of state variables of TGC$_i$, $\sum_{i=1}^{k} n_i = n$
m_i	Number of devices of TGC$_i$
$\mathcal{B}$	Set of boundary buses
$\mathcal{B}_i$	Set of boundary buses of TGC$_i$
$\mathcal{L}$	Set of inter-area lines
$\mathcal{D}_i$	Set of devices in TGC$_i$
$\mathcal{C}$	Set of controlled parties
$\mathsf{Enc}_i(\mu)$	Encryption of message μ using the public key of party i
$\mathsf{Dec}_i(\mu)$	Decryption of message μ using the secret key of party i
$\mathsf{Eval}_i(f)$	Homomorphic evaluation of function f using the evaluation key of party i
$\boldsymbol{x}$	State variables of the grid, $\boldsymbol{x} = [x_1, \cdots, x_n]^{\mathrm{T}}$
$\boldsymbol{x}_{\text{flat}}$	Flat voltage profile ($V_i = 1$ pu, $\theta_i = 0$)
$\boldsymbol{x}_i$	State variables of TGC$_i$, $\cup_{i=1}^{k} \boldsymbol{x}_i = \boldsymbol{x}$
$\hat{\boldsymbol{x}}^*$	Masked state estimates of the grid, $\hat{\boldsymbol{x}}^* = [\hat{x}_1^*, \cdots, \hat{x}_n^*]^{\mathrm{T}}$
$\hat{\boldsymbol{x}}_i^*$	Masked state estimates of TGC$_i$, $\cup_{i=1}^{k} \hat{\boldsymbol{x}}_i^* = \hat{\boldsymbol{x}}^*$
$\hat{\boldsymbol{x}}^{(t)}$	State estimates of the grid at the t-th iteration
$\boldsymbol{x}^{(in)}$	Internal state variables
$\boldsymbol{x}^{(bo)}$	Boundary state variables
$\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$	Estimates of partial derivatives of boundary measurement functions at $\hat{\boldsymbol{x}}^{(bo)(t)}$
$\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$	Estimates of boundary measurement functions at $\hat{\boldsymbol{x}}^{(bo)(t)}$
$\boldsymbol{H}_i(\hat{\boldsymbol{x}}_i^{(t)})$	Estimates of partial derivatives of TGC$_i$'s measurement functions at $\hat{\boldsymbol{x}}_i^{(t)}$
$\boldsymbol{h}_i(\hat{\boldsymbol{x}}_i^{(t)})$	Estimates of TGC$_i$'s measurement functions at $\hat{\boldsymbol{x}}_i^{(t)}$
T	The number of iterations until π_{AC} converges
view_i^{π}	View of adversary $\mathcal{A}_i$ in the protocol π
MAX	The maximum value of a state variable $x \in [-MAX, MAX]$
ε	Differential privacy parameter
τ	Convergence threshold of ACSE
$\boldsymbol{X} \stackrel{\text{c}}{\equiv} \boldsymbol{Y}$	$\boldsymbol{X}$ and $\boldsymbol{Y}$ are computationally indistinguishable

3.2 State estimation

State estimation for electric transmission grids was first formulated as a weighted least-squares problem in [3]. State estimation is a central and essential part of every power control system. The main function of state estimation is to perform computer analysis of grid states under the conditions characterized by a set of measurements. Specifically, the output of state estimation is the value of the system state vector at a specific time point of measurement reading. Most state estimation programs in practical use are formulated as over-determined systems of equations (i.e., systems with more equations than unknowns) and solved as weighted least-squares problems.

The relationship between the measurement data and the states can be represented as a vector function $\boldsymbol{h}(\cdot)$ relating measurements to states, which are linear functions in DC models or non-linear functions in AC models [28]:

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{e} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the true system state vector of an N-bus power system, $\boldsymbol{y} \in \mathbb{R}^m$ is the measurement vector, $\boldsymbol{e} \in \mathbb{R}^m$ is a measurement error vector assumed to be zero-mean Gaussian distributed. The corresponding variance matrix is denoted by $\boldsymbol{S} = \text{diag}(\sigma_1^2, \sigma_2^2, \cdots, \sigma_m^2)$, where σ_i^2 is the variance of reading errors of the i-th measuring device ($i \in \{1, \cdots, m\}$). Every two measuring devices are mutually independent.

The process of obtaining the estimated state vector $\hat{\boldsymbol{x}}$ is called state estimation which is considered the problem of minimizing the function:

$$J(\boldsymbol{x}) = \frac{1}{2} \sum_{i=1}^{m} \left(\frac{y_i - h_i(\boldsymbol{x})}{\sigma_i} \right)^2 \tag{2}$$

where the function $h_i(\boldsymbol{x})$ are the expressions of the measurements (e.g. power flows) in terms of states $\boldsymbol{x}$. Define a Jacobian matrix $\boldsymbol{H} \in \mathbb{R}^{m \times n}$ as:

$$\boldsymbol{H} = \frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial h_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial h_1(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial h_1(\boldsymbol{x})}{\partial x_n} \\ \frac{\partial h_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial h_2(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial h_2(\boldsymbol{x})}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_m(\boldsymbol{x})}{\partial x_1} & \frac{\partial h_m(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial h_m(\boldsymbol{x})}{\partial x_n} \end{bmatrix} \tag{3}$$

3.2.1 DC state estimation

In a DC model, the state variables are the angles. The linear relationship between the measurement data, $\boldsymbol{y}$, and the states, $\boldsymbol{x}$, is given by:

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{x} + \boldsymbol{e} \tag{4}$$

$$J(\boldsymbol{x}) = (\boldsymbol{y} - \boldsymbol{H} \cdot \boldsymbol{x})^T \cdot \boldsymbol{S}^{-1} \cdot (\boldsymbol{y} - \boldsymbol{H} \cdot \boldsymbol{x}) \tag{5}$$

where $\boldsymbol{x} = [\theta_2, \theta_3, \cdots, \theta_n]^T \in \mathbb{R}^{n-1}$ are bus voltage angles for a n-bus power grid.

The value of $\hat{\boldsymbol{x}}$ minimizing J satisfies the normal equation $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ where $\boldsymbol{A} = \boldsymbol{H}^T \cdot \boldsymbol{S}^{-1} \cdot \boldsymbol{H}$, $\boldsymbol{b} = \boldsymbol{H}^T \cdot \boldsymbol{S}^{-1} \cdot \boldsymbol{y}$. Thus, by solving the linear system equations $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$, the state estimate $\hat{\boldsymbol{x}}$ with regarding to a DC model is obtained [28]. Algorithm 1 (DCSE) presents DC state estimation.

- If a device d measures a power flow from bus i to bus j over the line ij connecting bus i to bus j, then:

$$h_d(\boldsymbol{x}) = b_{ij}(\theta_i - \theta_j) \tag{6}$$

where b_{ij} is the parameter of the line ij

Algorithm 1: DCSE($\boldsymbol{y}, \boldsymbol{R}, \mathcal{G}$) [28]

Result: State estimate $\hat{\boldsymbol{x}}$

1 Based on the DC network model of $\mathcal{G}$, form $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{H} \cdot \boldsymbol{x}$

2 Compute $\boldsymbol{A} = \boldsymbol{H}^T \cdot \boldsymbol{W} \cdot \boldsymbol{H}$, $\boldsymbol{b} = \boldsymbol{H}^T \cdot \boldsymbol{W} \cdot \boldsymbol{y}$, where $\boldsymbol{W} = \boldsymbol{S}^{-1}$

3 Solve $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ to obtain $\hat{\boldsymbol{x}}$

4 return $\hat{\boldsymbol{x}}$

3.2.2 AC state estimation

In an AC model, state variables are phase angles and voltage magnitudes. The relationship between the measurements (e.g., active power flows, reactive power flows) and the states forms a set of non-linear equations. The state variables in AC models are voltages magnitudes V_i and angles θ_i ($i \in \{1, \cdots, N\}$) of all buses in the power grid. The nonlinear relationship between the state variable, $\boldsymbol{x}$, and the measurement, $\boldsymbol{y}$, can be formulated as:

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{e}, \tag{7}$$

where $\boldsymbol{x} = [\theta_2, \theta_3, \cdots, \theta_n, V_1, V_2, \cdots, V_n]^T \in \mathbb{R}^{2n-1}$ and $\boldsymbol{h}(\boldsymbol{x})$ is the nonlinear function between the measurement data and the state variables.

- If a device d measures an active power flow from bus i to bus j over the line ij connecting bus i to bus j, then:

$$\begin{aligned} h_d(\boldsymbol{x}) &= f_{ij}^P(V_i, V_j, \theta_i, \theta_j) \\ &= V_i^2 g_{ij} - V_i V_j (g_{ij}\text{cos}\theta_{ij} + b_{ij}\text{sin}\theta_{ij}) \end{aligned} \tag{8}$$

$$h_d^* = h_d(\hat{\boldsymbol{x}}) = f_{ij}^P(V_i = \hat{V}_i, V_j = \hat{V}_j, \theta_i = \hat{\theta}_i, \theta_j = \hat{\theta}_j) \tag{9}$$

- If a device d measures an reactive power flow from bus i to bus j over the line ij connecting bus i to bus j, then:

$$\begin{aligned} h_d(\boldsymbol{x}) &= f_{ij}^Q(V_i, V_j, \theta_i, \theta_j) \\ &= -V_i^2(b_{ij} + b_{ij}^s) - V_i V_j (g_{ij}\text{sin}\theta_{ij} - b_{ij}\text{cos}\theta_{ij}); \end{aligned} \tag{10}$$

$$h_d^* = h_d(\hat{\boldsymbol{x}}) = f_{ij}^Q(V_i = \hat{V}_i, V_j = \hat{V}_j, \theta_i = \hat{\theta}_i, \theta_j = \hat{\theta}_j) \tag{11}$$

where $\theta_{ij} = \theta_i - \theta_j$ and $(g_{ij}, b_{ij}, b_{ij}^s)$ are the parameters of the line ij.

Consider these above measurements with $(V_i, V_j, \theta_i, \theta_j)$ as the state variables, the corresponding elements of a Jacobian matrix $\boldsymbol{H}$ of $\boldsymbol{h}$ are given as follows:

$$\boldsymbol{H} = \begin{bmatrix} \vdots & \vdots & \vdots & \vdots & \\ \cdots \ \dfrac{\partial f_{ij}^P}{\partial \theta_i} & \cdots \ \dfrac{\partial f_{ij}^P}{\partial \theta_j} & \cdots \ \dfrac{\partial f_{ij}^P}{\partial V_i} & \cdots \ \dfrac{\partial f_{ij}^P}{\partial V_j} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \\ \cdots \ \dfrac{\partial f_{ij}^Q}{\partial \theta_i} & \cdots \ \dfrac{\partial f_{ij}^Q}{\partial \theta_j} & \cdots \ \dfrac{\partial f_{ij}^Q}{\partial V_i} & \cdots \ \dfrac{\partial f_{ij}^Q}{\partial V_j} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{bmatrix} \tag{12}$$

$$\frac{\partial f_{ij}^P}{\partial V_i} = 2V_i g_{ij} - V_j g_{ij} cos\theta_{ij} - V_j b_{ij} sin\theta_{ij} \tag{13}$$

$$\frac{\partial f_{ij}^P}{\partial V_j} = -V_i g_{ij} cos\theta_{ij} - V_i b_{ij} sin\theta_{ij} \tag{14}$$

$$\frac{\partial f_{ij}^P}{\partial \theta_i} = V_i V_j g_{ij} sin\theta_{ij} - V_i V_j b_{ij} cos\theta_{ij} \tag{15}$$

$$\frac{\partial f_{ij}^P}{\partial \theta_j} = -V_i V_j g_{ij} sin\theta_{ij} + V_i V_j b_{ij} cos\theta_{ij} \tag{16}$$

$$\frac{\partial f_{ij}^Q}{\partial V_i} = -2V_i(b_{ij} + b_{ij}^s) + V_j b_{ij} cos\theta_{ij} - V_j g_{ij} sin\theta_{ij} \tag{17}$$

$$\frac{\partial f_{ij}^Q}{\partial V_j} = V_i b_{ij} cos\theta_{ij} - V_i g_{ij} sin\theta_{ij} \tag{18}$$

$$\frac{\partial f_{ij}^Q}{\partial \theta_i} = -V_i V_j b_{ij} sin\theta_{ij} - V_i V_j g_{ij} cos\theta_{ij} \tag{19}$$

$$\frac{\partial f_{ij}^Q}{\partial \theta_j} = V_i V_j b_{ij} sin\theta_{ij} + V_i V_j g_{ij} cos\theta_{ij} \tag{20}$$

3.3 Differential privacy

Differential privacy [29] is a strong privacy model that resists background attacks and provides a provable privacy guarantee. Even if an adversary knows the maximum background information such as all the other records in a data set except one record, differential privacy theoretically proves that there is a low probability of the adversary figuring out the unknown record.

Algorithm 2: ACSE($\boldsymbol{y}, \boldsymbol{R}, \mathcal{G}$) [28]

Result: State estimate $\hat{\boldsymbol{x}}$

1 Based on the AC network model of $\mathcal{G}$, form a vector function $\boldsymbol{h}(\boldsymbol{x})$ and $\boldsymbol{H}(\boldsymbol{x}) = \frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}}$

2 t = 0, Initialize $\hat{\boldsymbol{x}}^{(t)} \leftarrow \boldsymbol{x}_{\text{flat}}$

3 Compute $\overline{\boldsymbol{H}} = \boldsymbol{H}(\hat{\boldsymbol{x}}^{(t)}), \boldsymbol{h}^* = \boldsymbol{h}(\hat{\boldsymbol{x}}^{(t)})$, $\boldsymbol{A} = \overline{\boldsymbol{H}}^T \cdot \boldsymbol{W} \cdot \overline{\boldsymbol{H}}, \boldsymbol{b} = \overline{\boldsymbol{H}}^T \cdot \boldsymbol{W} \cdot (\boldsymbol{y} - \boldsymbol{h}^*)$ where $\boldsymbol{W} = \boldsymbol{S}^{-1}$

4 Solve $\boldsymbol{A} \cdot \Delta \boldsymbol{x} = \boldsymbol{b}$ to obtain $\Delta \hat{\boldsymbol{x}}^{(t)}$

5 $\hat{\boldsymbol{x}}^{(t+1)} = \hat{\boldsymbol{x}}^{(t)} + \Delta \hat{\boldsymbol{x}}^{(t)}$

6 If ($\|\Delta \hat{\boldsymbol{x}}^{(t)}\|_\infty > \tau$) then { $t = t + 1$; go to 3; }

7 else return $\hat{\boldsymbol{x}}^{(t)}$

A randomized mechanism $\mathcal{M}$ gives ε-differential privacy for every set of outputs $\mathcal{S}$, and for any neighbouring datasets of D and D', if $\mathcal{M}$ satisfies

$$\mathsf{Pr}[\mathcal{M}(D) \in \mathcal{S}] \leq \exp(\varepsilon) \cdot \mathsf{Pr}[\mathcal{M}(D') \in \mathcal{S}]$$

where $\mathsf{Pr}[\cdot]$ denotes probability and ε is the privacy budget. A smaller ε corresponds to stronger privacy protection, and vice versa.

Laplace Mechanism [30]: Given a function $f : D \to \mathbb{R}$ over a data set D, the following mechanism $\mathcal{M}$ provides the ε-differential privacy

$$\mathcal{M}(D) = f(D) + \text{Lap}(0, b)$$

where $\text{Lap}(0, b)$ is a random noise sampled from the Laplace distribution with mean $\mu = 0$, scale $b = \frac{\Delta f}{\varepsilon}$, and Δf is the sensitivity of the function f.

Differential privacy can be applied in the local setting where there is no trusted data aggregator and each user publishes the private data after adding noise individually [31], [32]. This local setting is for local differential privacy, which is a distributed variant of differential privacy [33]. The neighbouring datasets in local differential privacy are defined as two different values of the input domain. Randomized mechanisms satisfying differential privacy can also be applied in a distributed manner to achieve local differential privacy [33], [34], [35], [36], [37]. In a local differential privacy model, each user locally perturbs her data and then publishes the perturbed data to the server. This model provides strong protection because only the users know their exact data value. Given a noised output from a user, the original data is protected because all the possible values have similar probabilities to report the given perturbed output.

Local differential privacy [31], [33]: A randomized mechanism $\mathcal{M}$ satisfies ε-local differential privacy if and only if for any pairs of input values v and v' in the domain of $\mathcal{M}$, and for any possible output s in the range $\mathcal{R}$ of $\mathcal{M}$, it holds

$$\mathsf{Pr}[\mathcal{M}(v) = s] \leq \exp(\varepsilon) \cdot \mathsf{Pr}[\mathcal{M}(v') = s]$$

3.4 Homomorphic Encryption

Informally, homomorphic encryption is a type of encryption that allows a computation performed on ciphertexts to generate an encrypted result such that if it is decrypted, the computation result performed on the plain texts will be the same. Formally, a homomorphic public-key encryption scheme $\mathcal{E}$ with key space $\mathcal{K}$, message space $\mathcal{M}$, and ciphertext space $\mathcal{C}$ is composed of the following algorithms:

- $\mathcal{E}.\mathsf{KeyGeneration}(\lambda) \to \{sk, pk, ek\}$: given the security paramter λ, output a secret key sk, a public key pk, and an evaluation key ek.
- $\mathcal{E}.\mathsf{Encryption}(pk, \mu) \to c$: given the public key pk and a message $\mu \in \mathcal{M}$, output a ciphertext c.
- $\mathcal{E}.\mathsf{Decryption}(sk, c) \to \mu$: given the secret key sk and a ciphertext c, output $\mu \in \mathcal{M}$, where $c = \mathcal{E}.\mathsf{Encryption}(pk, \mu)$.
- $\mathcal{E}.\mathsf{Evaluation}(f, c_1, \cdots, c_l, ek) \to c_f$: given a function $f : \mathcal{M}^l \to \mathcal{M}$, l ciphertexts $c_i = \mathcal{E}.\mathsf{Encryption}(pk, \mu_i), i \in \{1, \cdots, l\}$ and an evaluation key ek, output a ciphertext c_f such that:

$$\mathcal{E}.\mathsf{Decryption}(sk, c_f) = f(\mu_1, \cdots, \mu_l)$$

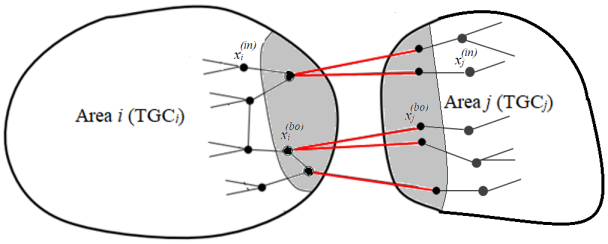

Fig. 1: Interconnection area between area i and area j

4 System Model and Threat Model

4.1 System Model

Consider an N-bus multi-area grid as k non-overlapping areas managed by k independent transmission grid companies (TGCs). If a bus in one sub-grid connects to buses in other sub-grids, it is a boundary bus; otherwise, it is an internal bus. States corresponding to boundary (internal) buses are called boundary (internal) states. Similarly, if a measurement in one sub-grid is relevant to state variables or line parameters of other sub-grids, it is a boundary measurement; otherwise, it is an internal measurement. Each TGC owns private data, including internal measurements, internal topology and line parameters, and a set of estimated states corresponding to its buses. There are two types of states (or buses) that belong to each TGC: internal and boundary states (buses). For measurements, each TGC controls its internal measurements. The boundary measurements (i.e. power flows along tie-lines) are processed by a system operator (SO). An SO manages the interconnection area, consisting of tie-lines ending at two boundary buses (Fig. 1). A state estimation service (SE) solves the state estimation problem to help determine the states of the whole grid. System-wide state estimates are delivered to the corresponding TGCs (i.e. TGC$_i$ receives $\hat{\boldsymbol{x}}_i$).

Assume that SO and each TGC$_i$ possesses a set of measuring devices $\mathcal{D}_i = \{d_{i_1}, d_{i_2}, \cdots, d_{i_{m_i}}\}$ which provides measurements $\boldsymbol{y}_i = [y_{d_{i_1}}, y_{d_{i_2}}, \cdots, y_{d_{i_{m_i}}}]^{\mathrm{T}}$ and the corresponding variance matrix $\boldsymbol{S}_i = \text{diag}(\sigma^2_{d_{i_1}}, \sigma^2_{d_{i_2}}, \cdots, \sigma^2_{d_{i_{m_i}}})$. Here $i = 0$ for SO and $i \in \{1, \cdots, k\}$ corresponds to one of k

TGCs, $\mathcal{D}_i \subset \mathcal{D} = \{1, 2, \cdots, m\}$, $\mathcal{D}_0 \cup \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_k = \mathcal{D}$, $\mathcal{D}_i \cap \mathcal{D}_j = \emptyset$ with $i \neq j$ and $i, j \in \{0, .., k\}$.

For $j = [0, \cdots, k]$, define:

$$\boldsymbol{h}_j(\boldsymbol{x}) = \begin{bmatrix} h_{d_{j_1}}(\boldsymbol{x}) \\ h_{d_{j_2}}(\boldsymbol{x}) \\ \cdots \\ h_{d_{j_{m_j}}}(\boldsymbol{x}) \end{bmatrix}$$

where $h_{d_{j_1}}(\boldsymbol{x}), h_{d_{j_2}}(\boldsymbol{x}), \ldots, h_{d_{j_{m_j}}}(\boldsymbol{x})$ are power flows (See Eq. 6, 8, 10).

TGC$_i$'s internal measurement $\boldsymbol{y}_i$ relates to the state vector $\boldsymbol{x}$ by the vector function $\boldsymbol{h}_i$ and the vector error $\boldsymbol{e}_i$ as $\boldsymbol{y}_i = \boldsymbol{h}_i(\boldsymbol{x}) + \boldsymbol{e}_i$. SO's boundary measurement $\boldsymbol{y}_0$ relates to the state vector $\boldsymbol{x}$ by the vector function $\boldsymbol{h}_0$ and the vector error $\boldsymbol{e}_0$ as $\boldsymbol{y}_0 = \boldsymbol{h}_0(\boldsymbol{x}) + \boldsymbol{e}_0$. From Eq. 6, 8, 10, note that $\boldsymbol{h}_i$ ($i \in \{1, \cdots, k\}$) just depends on states $\boldsymbol{x}_i$ of a single sub-grid while $\boldsymbol{h}_0$ depends on boundary states of multiple sub-grids, then:

$$\boldsymbol{y}_i = \boldsymbol{h}_i(\boldsymbol{x}_i) + \boldsymbol{e}_i, i \in \{1, \cdots, k\} \tag{21}$$

$$\boldsymbol{y}_0 = \boldsymbol{h}_0(\boldsymbol{x}^{(bo)}) + \boldsymbol{e}_0 \tag{22}$$

4.2 Threat Model

A semi-honest adversarial model is considered against the scheme. Adversaries are assumed to be semi-honest in the sense that they follow the protocol but can obtain available transcripts to learn extra information that should remain private (i.e., passive security). A good estimate of system-wide states supporting security operations and power management is a common interest of all parties; thus, it is reasonable that they are incentivised to follow the protocol to achieve the best output. However, some parties might be motivated to conspire with each other against a target party for some business benefits. For example, SE or SO may attempt to learn information about private data contributed by a target honest TGC since this data can potentially be sold to other TGCs managed by competitive commercial power rivals. In the above-proposed system model, a semi-honest adversary can be any party except for the target honest TGC$_h$. SE and SO are assumed not to collude but might conspire with other colluded semi-honest TGC$_c$ against the target honest TGC$_h$.

Moreover, it is assumed that private and authenticated peer-to-peer channels exist between parties so that the data transferred cannot be modified. This can be enforced in practice with the appropriate use of Digital Signatures and Certificate Authorities.

5 Proposed Privacy-Preserving State Estimation Scheme in Transmission Power Grids

The proposed privacy-preserving state estimation scheme can be seen as secure multi-party protocols run by $k + 2$ parties (k TGCs, SE and SO). This section presents secure multiparty computation protocols for privacy-preserving state estimation (DC and AC models). First of all, non-privacy-preserving multiparty DC and AC state estimation are introduced. Then, protocols of privacy-preserving DC state estimation (Fig. 2) and privacy-preserving AC state estimation (Fig. 3) are designed based on the integration of privacy-preserving methods into the non-privacy-preserving versions.

5.1 Non-privacy-preserving version of multiparty state estimation

In the non-privacy-preserving version of multiparty state estimation for DC (Algorithm 3 - MDCSE) and AC (Algorithm 4 - MACSE), the data input for state estimation process is partitioned and prepared by different parties according to the partition of a whole transmission grid into multiple sub-grids. The partition is compatible with the system model provided above.

In AC models, we have:

$$\boldsymbol{A} = \overline{\boldsymbol{H}}^{\mathrm{T}} \cdot \boldsymbol{W} \cdot \overline{\boldsymbol{H}} \in \mathbb{R}^{n \times n} \tag{23}$$

$$\boldsymbol{b} = \overline{\boldsymbol{H}}^{\mathrm{T}} \cdot \boldsymbol{W} \cdot (\boldsymbol{y} - \boldsymbol{h}^*) \in \mathbb{R}^n \tag{24}$$

where:

$$\overline{\boldsymbol{H}} = \boldsymbol{H}(\hat{\boldsymbol{x}}) = \begin{bmatrix} \overline{\boldsymbol{h}}_{d_1}^{\mathrm{T}} \\ \overline{\boldsymbol{h}}_{d_2}^{\mathrm{T}} \\ \cdots \\ \overline{\boldsymbol{h}}_{d_m}^{\mathrm{T}} \end{bmatrix} \in \mathbb{R}^{m \times n} \tag{25}$$

$$\overline{\boldsymbol{h}}_{d_i} = \begin{bmatrix} \frac{\partial h_{d_i}}{\partial x_1}(\hat{x}_1) \\ \frac{\partial h_{d_i}}{\partial x_2}(\hat{x}_2) \\ \cdots \\ \frac{\partial h_{d_i}}{\partial x_n}(\hat{x}_n) \end{bmatrix} \in \mathbb{R}^n,\ i = 1, \cdots, m \tag{26}$$

$$\boldsymbol{h}^* = \begin{bmatrix} h_{d_1}(\hat{\boldsymbol{x}}) \\ h_{d_2}(\hat{\boldsymbol{x}}) \\ \cdots \\ h_{d_m}(\hat{\boldsymbol{x}}) \end{bmatrix} \in \mathbb{R}^m \tag{27}$$

$$\boldsymbol{W} = \mathrm{diag}(\sigma_{d_1}^{-2}, \sigma_{d_2}^{-2}, \cdots \sigma_{d_m}^{-2}) \tag{28}$$

Thus:

$$\boldsymbol{A} = \sum_{d \in \mathcal{D}} \sigma_d^{-2} \cdot \overline{\boldsymbol{h}}_d \cdot \overline{\boldsymbol{h}}_d^{\mathrm{T}} \tag{29}$$

$$\boldsymbol{b} = \sum_{d \in \mathcal{D}} \sigma_d^{-2} \cdot \overline{\boldsymbol{h}}_d \cdot (y_d - h_d(\hat{\boldsymbol{x}})) \tag{30}$$

Then we have:

$$\begin{aligned} \boldsymbol{A}_0 + \boldsymbol{A}_1 + \cdots + \boldsymbol{A}_k &= \sum_{i=0}^{k} \sum_{d \in \mathcal{D}_i} \sigma_d^{-2} \cdot \overline{\boldsymbol{h}}_d \cdot \overline{\boldsymbol{h}}_d^{\mathrm{T}} \\ &= \sum_{d \in \mathcal{D}} \sigma_d^{-2} \cdot \overline{\boldsymbol{h}}_d \cdot \overline{\boldsymbol{h}}_d^{\mathrm{T}} \\ &= \boldsymbol{A} \end{aligned} \tag{31}$$

$$\begin{aligned} \boldsymbol{b}_0 + \boldsymbol{b}_1 + \cdots + \boldsymbol{b}_k &= \sum_{i=0}^{k} \sum_{d \in \mathcal{D}_i} \sigma_d^{-2} \cdot \overline{\boldsymbol{h}}_d \cdot (y_d - h_d(\hat{\boldsymbol{x}})) \\ &= \sum_{d \in \mathcal{D}} \sigma_d^{-2} \cdot \overline{\boldsymbol{h}}_d \cdot (y_d - h_d(\hat{\boldsymbol{x}})) \\ &= \boldsymbol{b} \end{aligned} \tag{32}$$

Similar results apply to DC models. Therefore, the correctness of Algorithm 3 and Algorithm 4 is guaranteed as the same as Algorithm 1 and Algorithm 2.

Algorithm 3: MDCSE($\{\boldsymbol{y}_i, \boldsymbol{S}_i, \mathcal{G}_i\}_{i\in\{0,\cdots k\}}$)

Result: State estimate $\hat{\boldsymbol{x}}$

1 Based on the DC network model of $\mathcal{G}_i$, each party i forms a linear function vector: $\boldsymbol{h}_i(\boldsymbol{x}) = \boldsymbol{H}_i \cdot \boldsymbol{x}$

2 Each party $i \in \{0, \cdots, k\}$ computes $\boldsymbol{A}_i = \boldsymbol{H}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot \boldsymbol{H}_i$, $\boldsymbol{b}_i = \boldsymbol{H}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot \boldsymbol{y}_i$, where $\boldsymbol{W}_i = \boldsymbol{S}_i^{-1}$, then sends $(\boldsymbol{A}_i, \boldsymbol{b}_i)$ to the server

3 The server solves $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ to obtain $\hat{\boldsymbol{x}}$, where $\boldsymbol{A} = \sum_{i=0}^{k} \boldsymbol{A}_i$ and $\boldsymbol{b} = \sum_{i=0}^{k} \boldsymbol{b}_i$

4 return $\hat{\boldsymbol{x}}$

Algorithm 4: MACSE($\{\boldsymbol{y}_i, \boldsymbol{S}_i, \mathcal{G}_i\}_{i\in\{0,\cdots k\}}$)

Result: State estimate $\hat{\boldsymbol{x}}$

1 Based on the AC network model of $\mathcal{G}_i$, each party i forms a non-linear function vector $\boldsymbol{h}_i(\boldsymbol{x})$ and $\boldsymbol{H}_i(\boldsymbol{x}) = \frac{\partial \boldsymbol{h}_i(\boldsymbol{x})}{\partial \boldsymbol{x}}$

2 t = 1, Initialize $\hat{\boldsymbol{x}}^{(t)} \leftarrow \boldsymbol{x}_{\text{flat}}$

3 Each party $i \in \{0, \cdots, k\}$ computes

$$\overline{\boldsymbol{H}}_i = \boldsymbol{H}_i(\hat{\boldsymbol{x}}_i^{(t)}), \boldsymbol{h}_i^* = \boldsymbol{h}_i(\hat{\boldsymbol{x}}_i^{(t)}) \quad \text{if } i \in \{1, \cdots, k\}$$

$$\overline{\boldsymbol{H}}_i = \boldsymbol{H}_i(\hat{\boldsymbol{x}}^{(bo)(t)}), \boldsymbol{h}_i^* = \boldsymbol{h}_i(\hat{\boldsymbol{x}}^{(bo)(t)}) \quad \text{if } i = 0$$

$$\text{and } \boldsymbol{A}_i = \overline{\boldsymbol{H}}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot \overline{\boldsymbol{H}}_i, \boldsymbol{b}_i = \overline{\boldsymbol{H}}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot (\boldsymbol{y}_i - \boldsymbol{h}_i^*)$$

where $\boldsymbol{W}_i = \boldsymbol{S}_i^{-1}$, then sends $(\boldsymbol{A}_i, \boldsymbol{b}_i)$ to a server

4 The server solves $\boldsymbol{A} \cdot \Delta\boldsymbol{x} = \boldsymbol{b}$ to obtain $\Delta\hat{\boldsymbol{x}}^{(t)}$ where $\boldsymbol{A} = \sum_{i=0}^{k} \boldsymbol{A}_i$ and $\boldsymbol{b} = \sum_{i=0}^{k} \boldsymbol{b}_i$.

5 The server calculates $\hat{\boldsymbol{x}}^{(t+1)} = \hat{\boldsymbol{x}}^{(t)} + \Delta\hat{\boldsymbol{x}}^{(t)}$

6 If ($\|\Delta\hat{\boldsymbol{x}}^{(t)}\|_\infty > \tau$) and ($t < T_{max}$) then { $t = t + 1$; go to 3; }

7 else if ($\|\Delta\hat{\boldsymbol{x}}^{(t)}\|_\infty \leq \tau$) return ($sol = 1, \hat{\boldsymbol{x}}^{(t)}$)

8 else return $sol = 0$

5.2 Privacy-preserving state estimation schemes in transmission power grids

A commonly proposed approach is applied to construct privacy-preserving versions of multiparty state estimation for both DC and AC models. Each party encrypts/perturbs data before contributing their data for state estimation. The data protection process in both schemes includes data preparation, encryption, and masking. For both DC and AC models, the *Setup* procedure establishes cryptographic keys of $k+1$ parties (k TGCs and SE) to be used during the execution. Each party generates keys of the homomorphic encryption scheme, including a secret value key sk for decryption, a public key pk for encryption, and an evaluation key ek for homomorphic evaluation from algorithm $\mathsf{KeyGen}(\lambda)$, where λ is the security parameter of the homomorphic encryption scheme. pk and ek of TGCs are published to SE and SO. pk and ek of SE are published to TGCs and SO. The followings are the descriptions of privacy-preserving DC state estimation, as illustrated in Fig. 2, and privacy-preserving AC state estimation, as described in Fig. 3.

5.2.1 Privacy-preserving DC state estimation

For a DC state estimator (Fig. 2), because of the linearity property (see Eq. (4)), the matrix $\boldsymbol{H}$ is a constant matrix, then $\boldsymbol{H}_i$ ($\boldsymbol{H}_0$) is directly constructed by a single TGC$_i$ (SO).

Each TGC$_i$ prepares its private data $\boldsymbol{A}_i \in \mathbb{R}^{n\times n}$, $\boldsymbol{b}_i \in \mathbb{R}^n$. SO also prepares its $\boldsymbol{A}_0, \boldsymbol{b}_0$. For $i \in \{0, \cdots, k\}$:

$$\boldsymbol{A}_i = \boldsymbol{H}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot \boldsymbol{H}_i, \tag{33}$$

$$\boldsymbol{b}_i = \boldsymbol{H}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot \boldsymbol{y}_i \tag{34}$$

$\boldsymbol{A}_i$, $\boldsymbol{b}_i$ are encrypted by TGC$_i$ using the SE's public key:

$$\boldsymbol{EA}_i = \mathsf{Enc}_{\text{SE}}(\boldsymbol{A}_i), \tag{35}$$

$$\boldsymbol{eb}_i = \mathsf{Enc}_{\text{SE}}(\boldsymbol{b}_i) \tag{36}$$

Each TGC$_i$ sends their encrypted data $(\boldsymbol{EA}_i, \boldsymbol{eb}_i)$ to SO. When obtaining all data from k parties, SO homomorphically evaluates the function $\sum_{i=0}^{k} \boldsymbol{A}_i$ ($\sum_{i=0}^{k} \boldsymbol{b}_i$), given k ciphertexts $\boldsymbol{EA}_i$ (resp. $\boldsymbol{eb}_i$):

$$\boldsymbol{EA} = \mathsf{Eval}_{\text{SE}}(\sum_{i=0}^{k} \boldsymbol{A}_i) \tag{37}$$

$$\boldsymbol{eb} = \mathsf{Eval}_{\text{SE}}(\sum_{i=0}^{k} \boldsymbol{b}_i) \tag{38}$$

SO generates an invertible matrix $\boldsymbol{R} \leftarrow_{\$} [-1, 1]^{n\times n}$, a random vector $\boldsymbol{r}$ from Laplace distribution with zero mean and variance $v = \frac{2\cdot \text{MAX}}{\varepsilon}$ where MAX is the maximum value of a state variable.

$\boldsymbol{R}$ and $\boldsymbol{r}$ are used to homomorphically mask the value of $\boldsymbol{A}$ and $\boldsymbol{b}$, given the ciphertexts $\boldsymbol{EA}, \boldsymbol{eb}$:

$$\boldsymbol{EA}^* = \mathsf{Eval}_{\text{SE}}(\boldsymbol{R} \cdot \boldsymbol{A}) \tag{39}$$

$$\boldsymbol{eb}^* = \mathsf{Eval}_{\text{SE}}(\boldsymbol{R} \cdot (\boldsymbol{b} + \boldsymbol{A} \cdot \boldsymbol{r})) \tag{40}$$

Then SO sends $(\boldsymbol{EA}^*, \boldsymbol{eb}^*)$ to SE. After receiving $(\boldsymbol{EA}^*, \boldsymbol{eb}^*)$ from SO, SE uses its secret key to decrypt:

$$\boldsymbol{A}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{EA}^*) \tag{41}$$

$$\boldsymbol{b}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{eb}^*) \tag{42}$$

By solving the problem $\boldsymbol{A}^* \cdot \boldsymbol{x}^* = \boldsymbol{b}^*$, SE obtains $\hat{\boldsymbol{x}}^* = \begin{bmatrix} \hat{\boldsymbol{x}}_1^* \\ \vdots \\ \hat{\boldsymbol{x}}_k^* \end{bmatrix} \in \mathbb{R}^n$ which is the masked version of the estimated state vector $\hat{\boldsymbol{x}} = \begin{bmatrix} \hat{\boldsymbol{x}}_1 \\ \vdots \\ \hat{\boldsymbol{x}}_k \end{bmatrix} \in \mathbb{R}^n$. Next, SE encrypts the masked states $\hat{\boldsymbol{x}}_i^*$ of TGC$_i$ using TGC$_i$'s public key and then sends these ciphertexts to SO.

$$\boldsymbol{e}\hat{\boldsymbol{x}}_i^* = \mathsf{Enc}_i(\hat{\boldsymbol{x}}_i^*) \tag{43}$$

SO receives $\boldsymbol{e}\hat{\boldsymbol{x}}^* = \begin{bmatrix} \boldsymbol{e}\hat{\boldsymbol{x}}_1^* \\ \vdots \\ \boldsymbol{e}\hat{\boldsymbol{x}}_k^* \end{bmatrix}$ and homomorphically evaluates:

$$\boldsymbol{e}\hat{\boldsymbol{x}}_i = \mathsf{Eval}_i(\hat{\boldsymbol{x}}_i^* - \boldsymbol{r}_i) \tag{44}$$

Then SO sends $\boldsymbol{e}\hat{\boldsymbol{x}}_i$ to TGC$_i$. Receiving $\boldsymbol{e}\hat{\boldsymbol{x}}_i$ from SO, party TGC$_i$ uses its secret key for decryption to get the estimated state $\hat{\boldsymbol{x}}_i$:

$$\hat{\boldsymbol{x}}_i = \mathsf{Dec}_i(\boldsymbol{e}\hat{\boldsymbol{x}}_i) \tag{45}$$

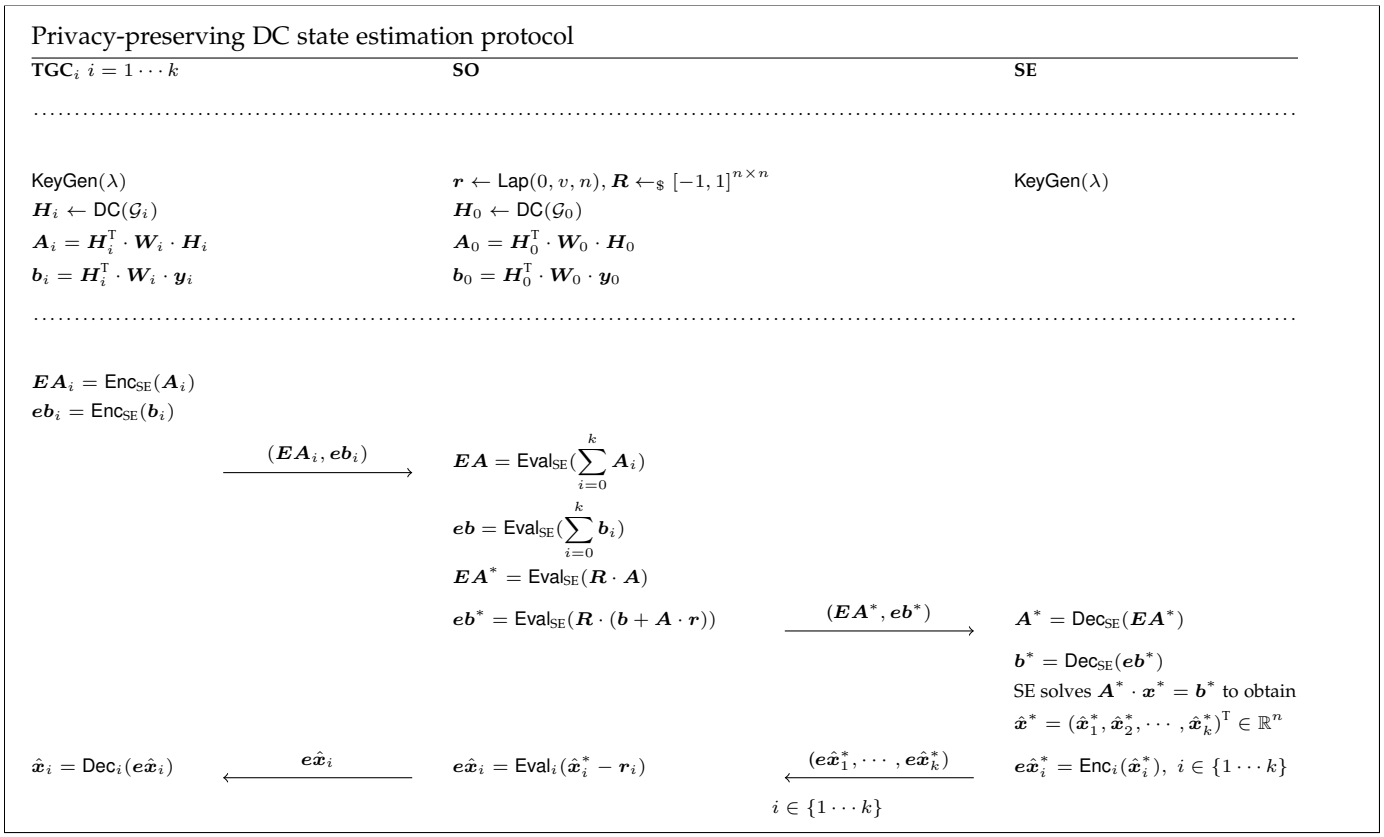

Fig. 2: Privacy-preserving DC state estimation

5.2.2 Privacy-preserving AC state estimation

For an AC state estimator (Fig. 3), $\boldsymbol{H}(\boldsymbol{x}) = \frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}}$ is a function of the states and changes its value based on the estimated states of the previous iteration. A flat voltage profile $\boldsymbol{x}_{\text{flat}}$ ($V_i = 1, \theta_i = 0$) can be used as the initialization of state estimates $\hat{\boldsymbol{x}}^{(1)}$ ($t = 1$).

It can be seen that each TGC$_i$ can compute $\boldsymbol{h}_i(\hat{\boldsymbol{x}}_i^{(t)}), \boldsymbol{H}_i(\hat{\boldsymbol{x}}_i^{(t)})$ with its tie-line parameters and state estimates $\hat{\boldsymbol{x}}_i^{(t)}$ from the previous iteration (line 3 of Algorithm 2). However, SO cannot calculate $\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ and $\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ by itself because it only has the boundary tie-line parameters, not the boundary state estimates $\hat{\boldsymbol{x}}^{(bo)(t)}$. Fortunately, homormophic encryption helps to solve this problem by allowing SO to homomorphically compute the encryptions of $\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ and $\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ based on the encryption of several specific functions of the boundary state estimates $\hat{\boldsymbol{x}}_i^{(bo)(t)}$ shared by each TGC$_i$.

Each TGC$_i$ prepares its private data $\boldsymbol{A}_i \in \mathbb{R}^{n \times n}$, $\boldsymbol{b}_i \in \mathbb{R}^n$:

$$\boldsymbol{A}_i = \overline{\boldsymbol{H}}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot \overline{\boldsymbol{H}}_i, \tag{46}$$

$$\boldsymbol{b}_i = \overline{\boldsymbol{H}}_i^{\mathrm{T}} \cdot \boldsymbol{W}_i \cdot (\boldsymbol{y}_i - \boldsymbol{h}_i^*) \tag{47}$$

$\boldsymbol{A}_0$ and $\boldsymbol{b}_0$ are available to SO in DC state estimation, but they are not in AC state estimation. What SO can have is the encryption $\boldsymbol{EA}_0$ of $\boldsymbol{A}_0$ and the encryption $\boldsymbol{eb}_0$ of $\boldsymbol{b}_0$.

TGC$_i$ sends their encrypted data $(\boldsymbol{EA}_i, \boldsymbol{eb}_i)$ to SO. Besides, for each boundary bus b in sub-grid $\mathcal{G}_i$, TGC$_i$ sends a vector of six ciphertexts $\boldsymbol{c}_b = (c_{b_1}, c_{b_2}, c_{b_3}, c_{b_4}, c_{b_5}, c_{b_6})$ which are encryptions of $\boldsymbol{p}_b = (V_b, \sin\theta_b, \cos\theta_b, V_b^2, V_b\sin\theta_b, V_b\cos\theta_b)$ to SO. SO homomorphically computes encryption of $\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ and $\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ from these ciphertexts received from all TGCs. Next, SO sends the encryptions of $\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ and $\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ to SE, who decrypts them to get $(\overline{\boldsymbol{H}}_0, \boldsymbol{h}_0^*)$ and calculates $(\boldsymbol{A}_0, \boldsymbol{b}_0)$. Then SE creates the ciphertexts $(\boldsymbol{EA}_0, \boldsymbol{eb}_0)$ which are sent back to SO.

Each TGC$_i$ has an indicator $\beta_i^{(t)}$, whose value 1 or 0 corresponds to the true or false of the condition $\|\Delta\hat{\boldsymbol{x}}_i^{(t)}\|_\infty > \tau$. The encryption of the indicator value $\beta_i^{(t)}$ helps hide the information that whether $\|\Delta\hat{\boldsymbol{x}}_i^{(t)}\|_\infty$ is greater than τ or not. The homomorphic encryption scheme is adopted again to encrypt $\beta_i^{(t)}$ under the SE's public key. Then, the encryption of the sum $\sum_{i=1}^{k} \beta_i^{(t)}$ is homomorphically evaluated by SO and then decrypted by SE to obtain the value of $cont^{(t)}$ which is sent back to all TGCs to instruct the loop to terminate or continue.

$$e\beta_i^{(t)} = \mathsf{Enc}_{\mathrm{SE}}(\beta_i^{(t)}) \tag{48}$$

$$econt^{(t)} = \mathsf{Enc}_{\mathrm{SE}}(\sum_{i=1}^{k} \beta_i^{(t)}) \tag{49}$$

$$cont^{(t)} = \mathsf{Dec}_{\mathrm{SE}}(econt^{(t)}) \tag{50}$$

Similar to privacy-preserving DC state estimation, having encrypted data from all k TGCs, SO homomorphically evaluates the sum function and the transformation using the random noises $(\boldsymbol{R}, \boldsymbol{r})$ it freshly generated at each iteration before sending the encrypted results to SE.

Privacy-preserving AC state estimation protocol: π_{AC}

	$\mathbf{TGC}_i$ $i = 1 \cdots k$		**SO**		**SE**
INIT	$\mathsf{KeyGen}(\lambda)$				$\mathsf{KeyGen}(\lambda)$
	$(\boldsymbol{H}_i(\boldsymbol{x}_i), \boldsymbol{h}_i(\boldsymbol{x}_i)) \leftarrow \mathsf{AC}(\mathcal{G}_i)$		$(\boldsymbol{H}_0(\boldsymbol{x}^{(bo)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)})) \leftarrow \mathsf{AC}(\mathcal{G}_0)$		
$t = 1$	$\hat{\boldsymbol{x}}^{(1)} \leftarrow \boldsymbol{x}_{\text{flat}}$				
REPEAT					
	$\overline{\boldsymbol{H}}_i = \boldsymbol{H}_i(\hat{\boldsymbol{x}}_i^{(t)}), \boldsymbol{h}_i^* = \boldsymbol{h}_i(\hat{\boldsymbol{x}}_i^{(t)})$				
	$\boldsymbol{A}_i = \overline{\boldsymbol{H}}_i^{\mathsf{T}} \cdot \boldsymbol{W}_i \cdot \overline{\boldsymbol{H}}_i$				
	$\boldsymbol{b}_i = \overline{\boldsymbol{H}}_i^{\mathsf{T}} \cdot \boldsymbol{W}_i \cdot (\boldsymbol{y}_i - \boldsymbol{h}_i^*)$				
	$\boldsymbol{EA}_i = \mathsf{Enc}_{\text{SE}}(\boldsymbol{A}_i), \boldsymbol{eb}_i = \mathsf{Enc}_{\text{SE}}(\boldsymbol{b}_i)$				
	$\boldsymbol{c}_b = \mathsf{Enc}_{\text{SE}}(\boldsymbol{p}_b), b \in \mathcal{B}_i$	$(\boldsymbol{EA}_i, \boldsymbol{eb}_i), \{\boldsymbol{c}_b\} \xrightarrow{\quad}$	$\boldsymbol{r} \leftarrow \mathsf{Lap}(0, v, n), \boldsymbol{R} \leftarrow_{\$} [-1, 1]^{n \times n}$		
			$\boldsymbol{E}\overline{\boldsymbol{H}}_0 = \mathsf{Eval}_{\text{SE}}(\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)}))$		
			$\boldsymbol{eh}_0^* = \mathsf{Eval}_{\text{SE}}(\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)}))$	$\xrightarrow{(\boldsymbol{E}\overline{\boldsymbol{H}}_0, \boldsymbol{eh}_0^*)}$	$\overline{\boldsymbol{H}}_0 = \mathsf{Dec}_{\text{SE}}(\boldsymbol{E}\overline{\boldsymbol{H}}_0), \boldsymbol{h}_0^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{eh}_0^*)$
					$\boldsymbol{EA}_0 = \mathsf{Enc}_{\text{SE}}(\overline{\boldsymbol{H}}_0^{\mathsf{T}} \cdot \boldsymbol{W}_0 \cdot \overline{\boldsymbol{H}}_0)$
			$\boldsymbol{EA} = \mathsf{Eval}_{\text{SE}}(\sum_{i=0}^{k} \boldsymbol{A}_i), \boldsymbol{eb} = \mathsf{Eval}_{\text{SE}}(\sum_{i=0}^{k} \boldsymbol{b}_i)$	$\xleftarrow{(\boldsymbol{EA}_0, \boldsymbol{eb}_0)}$	$\boldsymbol{eb}_0 = \mathsf{Enc}_{\text{SE}}(\overline{\boldsymbol{H}}_0^{\mathsf{T}} \cdot \boldsymbol{W}_0 \cdot (\boldsymbol{y}_0 - \boldsymbol{h}_0^*))$
			$\boldsymbol{EA}^* = \mathsf{Eval}_{\text{SE}}(\boldsymbol{R} \cdot \boldsymbol{A})$		
			$\boldsymbol{eb}^* = \mathsf{Eval}_{\text{SE}}(\boldsymbol{R} \cdot (\boldsymbol{b} + \boldsymbol{A} \cdot \boldsymbol{r}))$	$\xrightarrow{(\boldsymbol{EA}^*, \boldsymbol{eb}^*)}$	$\boldsymbol{A}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{EA}^*), \boldsymbol{b}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{eb}^*)$
					Solves $\boldsymbol{A}^* \cdot \Delta\boldsymbol{x}^* = \boldsymbol{b}^*$ to obtain $\Delta\hat{\boldsymbol{x}}^{*(t)}$
	$\Delta\hat{\boldsymbol{x}}_i^{(t)} = \mathsf{Dec}_i(\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_i^{(t)})$	$\xleftarrow{\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_i^{(t)}}$	$\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_i^{(t)} = \mathsf{Eval}_i(\Delta\hat{\boldsymbol{x}}_i^{*(t)} - \boldsymbol{r}_i)$	$\xleftarrow{(\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_1^{*(t)}, \cdots, \boldsymbol{e}\Delta\hat{\boldsymbol{x}}_k^{*(t)})}$	$\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_i^{*(t)} = \mathsf{Enc}_i(\Delta\hat{\boldsymbol{x}}_i^{*(t)})$
	$\hat{\boldsymbol{x}}_i^{(t+1)} = \hat{\boldsymbol{x}}_i^{(t)} + \Delta\hat{\boldsymbol{x}}_i^{(t)}$		$i \in \{1 \cdots k\}$		
	if $(\lVert\Delta\hat{\boldsymbol{x}}_i^{(t)}\rVert_\infty > \tau)$ $\beta_i^{(t)} = 1$; else $\beta_i^{(t)} = 0$;				
	$e\beta_i^{(t)} = \mathsf{Enc}_{\text{SE}}(\beta_i^{(t)})$	$\xrightarrow{e\beta_i^{(t)}}$	$econt^{(t)} = \mathsf{Eval}_{\text{SE}}(\sum \beta_i^{(t)})$	$\xrightarrow{econt^{(t)}}$	
					$cont^{(t)} = \mathsf{Dec}_{\text{SE}}(econt^{(t)})$
	$t = t + 1$	$\xleftarrow{cont^{(t)}}$		$\xleftarrow{cont^{(t)}}$	
UNTIL	$(cont^{(t-1)} = 0) \lvert\lvert (t = T_{max} + 1)$				

Fig. 3: Privacy-preserving AC state estimation

SE decrypts the ciphertexts received from SO and then solves the problem $\boldsymbol{A}^* \cdot \Delta\boldsymbol{x}^* = \boldsymbol{b}^*$ to obtain $\Delta\hat{\boldsymbol{x}}^{*(t)}$. Then, SE encrypts each $\Delta\hat{\boldsymbol{x}}_i^{*(t)}$ using the public key of the corresponding TGC_i, and sends all these ciphertexts to SO. Receiving these ciphertexts from SE, SO carries out homomorphic computation on the encrypted data to obtain the encryption of $\Delta\hat{\boldsymbol{x}}_i^{(t)}$ under the public key of the corresponding TGC_i. These ciphertexts are finally sent back to TGC_i. Using the secret key, TGC_i obtains $\Delta\hat{\boldsymbol{x}}_i^{(t)}$ and the state estimates for the next iteration $\hat{\boldsymbol{x}}_i^{(t+1)} = \hat{\boldsymbol{x}}_i^{(t)} + \Delta\hat{\boldsymbol{x}}_i^{(t)}$

6 Analysis of the Proposed Schemes

6.1 Correctness

6.1.1 Privacy-preserving DC state estimation

Based on the correctness of the homomorphic evaluation of the underlying homomorphic encryption scheme, from Eq. (39), (40), we have:

$$\boldsymbol{A}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{E}\boldsymbol{A}^*) = \boldsymbol{R} \cdot \boldsymbol{A} \tag{51}$$

$$\boldsymbol{b}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{e}\boldsymbol{b}^*) = \boldsymbol{R} \cdot (\boldsymbol{b} + \boldsymbol{A} \cdot \boldsymbol{r}) \tag{52}$$

From Eq. (44), (45):

$$\hat{\boldsymbol{x}}_i = \mathsf{Dec}_i(\boldsymbol{e}\hat{\boldsymbol{x}}_i) = \hat{\boldsymbol{x}}_i^* - \boldsymbol{r}_i \tag{53}$$

$$\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}^* - \boldsymbol{r} \tag{54}$$

Multiply both sides of Eq. (54) with the matrix $\boldsymbol{A}$, we have:

$$\begin{aligned}
\boldsymbol{A} \cdot \hat{\boldsymbol{x}} &= \boldsymbol{A} \cdot (\hat{\boldsymbol{x}}^* - \boldsymbol{r}) \\
&= \boldsymbol{R}^{-1} \cdot \boldsymbol{R} \cdot \boldsymbol{A} \cdot (\hat{\boldsymbol{x}}^* - \boldsymbol{r}) \\
&= \boldsymbol{R}^{-1} \cdot (\boldsymbol{A}^* \cdot \hat{\boldsymbol{x}}^* - \boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{r}) \quad \text{from Eq. (51)} \\
&= \boldsymbol{R}^{-1} \cdot (\boldsymbol{b}^* - \boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{r}) \\
&= \boldsymbol{R}^{-1} \cdot \boldsymbol{R} \cdot \boldsymbol{b} \quad \text{from Eq. (52)} \\
&= \boldsymbol{b}
\end{aligned}$$

$\boldsymbol{A} \cdot \hat{\boldsymbol{x}} = \boldsymbol{b}$ shows that $\hat{\boldsymbol{x}}$ is the state estimates of non-privacy-preserving state estimation.

6.1.2 Privacy-preserving AC state estimation

Note that protocol π_{AC} is a sequential composition of INIT procedure and T iterations of the function block within the Repeat-Until loop (Fig. 3), which is denoted as π_t.

$$\pi_{\text{AC}} = \text{INIT} \| \{\pi_t\}_{t=1}^{T}$$

To demonstrate that π_{AC} is correct, we prove that π_t correctly estimates $\Delta\hat{\boldsymbol{x}}$ and the convergence condition $\|\Delta\hat{\boldsymbol{x}}\|_\infty < \tau$ is correctly checked at the t-th iteration.

As can be seen from Fig. 2 and Fig. 3, $\boldsymbol{y} - \boldsymbol{h}^*$ and $\Delta\hat{\boldsymbol{x}}$ in π_t substitutes $\boldsymbol{y}$ and $\hat{\boldsymbol{x}}$ in π_{DC} respectively. The changes between π_{DC} and π_t is that $(\overline{\boldsymbol{H}}_0, \boldsymbol{h}_0^*)$ (or $(\boldsymbol{A}_0, \boldsymbol{b}_0)$) are not available for party SO in π_t. Thus, in π_t SO needs to homomorphically evaluate functions $\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ and $\boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$ to get the encryptions of $(\overline{\boldsymbol{H}}_0, \boldsymbol{h}_0^*)$, and then the encryption of $(\boldsymbol{A}_0, \boldsymbol{b}_0)$. The correctness of homomorphic evaluation of the underlying homomorphic encryption scheme guarantees that SO obtains the correct encryption of $(\boldsymbol{A}_0, \boldsymbol{b}_0)$. Consequently, π_t correctly estimates $\Delta\hat{\boldsymbol{x}}^{(t)}$ at the t-th iteration.

$$\boldsymbol{b}^* = \mathsf{Dec}_{\text{SE}}(\boldsymbol{e}\boldsymbol{b}^*) = \boldsymbol{R} \cdot (\boldsymbol{b} + \boldsymbol{A} \cdot \boldsymbol{r}) \tag{55}$$

$$\Delta\hat{\boldsymbol{x}}_i^{(t)} = \mathsf{Dec}_i(\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_i^{(t)}) = \Delta\hat{\boldsymbol{x}}_i^{*(t)} - \boldsymbol{r}_i \tag{56}$$

$$\Delta\hat{\boldsymbol{x}}^{(t)} = \Delta\hat{\boldsymbol{x}}^{*(t)} - \boldsymbol{r} \tag{57}$$

Multiply both sides of Eq. (57) with the matrix $\boldsymbol{A}$, we have:

$$\begin{aligned}
\boldsymbol{A} \cdot \Delta\hat{\boldsymbol{x}}^{(t)} &= \boldsymbol{A} \cdot (\Delta\hat{\boldsymbol{x}}^{*(t)} - \boldsymbol{r}) \\
&= \boldsymbol{R}^{-1} \cdot \boldsymbol{R} \cdot \boldsymbol{A} \cdot (\Delta\hat{\boldsymbol{x}}^{*(t)} - \boldsymbol{r}) \\
&= \boldsymbol{R}^{-1} \cdot (\boldsymbol{A}^* \cdot \Delta\hat{\boldsymbol{x}}^{*(t)} - \boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{r}) \quad \text{from Eq. (51)} \\
&= \boldsymbol{R}^{-1} \cdot (\boldsymbol{b}^* - \boldsymbol{R} \cdot \boldsymbol{A} \cdot \boldsymbol{r}) \\
&= \boldsymbol{R}^{-1} \cdot \boldsymbol{R} \cdot \boldsymbol{b} \quad \text{from Eq. (55)} \\
&= \boldsymbol{b}
\end{aligned}$$

The convergence condition is checked correctly. In fact, the loop terminates when:

$$\begin{aligned}
\|\Delta\hat{\boldsymbol{x}}^{(t)}\|_\infty \leq \tau &\equiv \{\|\Delta\hat{\mathbf{x}}_i^{(t)}\|_\infty \leq \tau\}_{i=1\cdots k} \\
&\equiv \{\beta_i^{(t)} = 0\}_{i=1\cdots k} \\
&\equiv \sum_{i=1}^{k} \beta_i^{(t)} = 0 \\
&\equiv cont^{(t)} = 0
\end{aligned}$$

The homomorphic encryption scheme adopted guarantees that $cont^{(t)} = \sum_{i=1}^{k} \beta_i^{(t)}$.

6.2 Privacy

To implement global state estimation, it necessitates sharing private data between TGCs and SO which violates TGCs' privacy. There are three different types of private data corresponding to each local TGC_i, which are the meter measurements ($\boldsymbol{y}_i$), internal line parameters ($\boldsymbol{h}_i(\boldsymbol{x}_i)$), and the estimated states ($\hat{\boldsymbol{x}}_i$). Here ($\boldsymbol{y}_i$, $\boldsymbol{h}_i(\boldsymbol{x}_i)$) is the private input and $\hat{\boldsymbol{x}}_i$ is the private output of state estimation. The following analyses the privacy protection that the schemes provide in a semi-honest adversarial model, with regarding to two types of adversaries:

- Semantic security protection of the private input and output against an adversary $\mathcal{A}_1$ controlling SO and colluded parties TGCs;
- Local differential privacy protection of the private output and multiplicative masking protection of the private input against an adversary $\mathcal{A}_2$ controlling SE and colluded parties TGCs.

6.2.1 Privacy protection against an adversary $\mathcal{A}_1$

In a semi-honest adversarial model, the adversary $\mathcal{A}_1$ has to follow exactly the protocol; thus, the leakage of private information of an honest party is only attributed to the view of $\mathcal{A}_1$ in the execution of the protocol. Therefore, the proof of privacy is based on the construction of a simulator who resides in a secure-by-definition "ideal world" and generates a view for $\mathcal{A}_1$ given $\mathcal{A}_1$'s input and output. The requirement is that the generated view is computationally indistinguishable from the real view of $\mathcal{A}_1$ in the "real world" (i.e. real execution of the protocol) [38]. This implies

TABLE 2: Encryption of $\boldsymbol{H}_0(\hat{\boldsymbol{x}}^{(bo)(t)}), \boldsymbol{h}_0(\hat{\boldsymbol{x}}^{(bo)(t)})$

$c_{b_1} = \text{Enc}_{\text{SE}}(V_b)$ $c_{b_2} = \text{Enc}_{\text{SE}}(\sin\theta_b)$	$c_{b_3} = \text{Enc}_{\text{SE}}(\cos\theta_b)$ $c_{b_4} = \text{Enc}_{\text{SE}}(V_b^2)$	$c_{b_5} = \text{Enc}_{\text{SE}}(V_b \cdot \sin\theta_b)$ $c_{b_6} = \text{Enc}_{\text{SE}}(V_b \cdot \cos\theta_b)$
$\text{Enc}_{\text{SE}}(h_{ij}^{*P}) = c_{i_4} \cdot g_{ij} - g_{ij} \cdot (c_{i_6} \cdot c_{j6} + c_{i_5} \cdot c_{j_5}) - b_{ij} \cdot (c_{i_5} \cdot c_{j6} - c_{j_5} \cdot c_{i_6})$		
$\text{Enc}_{\text{SE}}(h_{ij}^{*Q}) = -c_{i_4} \cdot (b_{ij} + b_{ij}^s) - g_{ij} \cdot (c_{i_5} \cdot c_{j_6} - c_{j_5} \cdot c_{i_6}) - b_{ij} \cdot (c_{i_6} \cdot c_{j_6} + c_{i_5} \cdot c_{j_5})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^P}{\partial V_i}) = 2c_{i_1} \cdot g_{ij} - g_{ij} \cdot (c_{j6} \cdot c_{i_3} + c_{j_5} \cdot c_{i_2}) - b_{ij} \cdot (c_{j_6} \cdot c_{i_2} - c_{j_5} \cdot c_{i_3})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^P}{\partial V_j}) = -g_{ij} \cdot (c_{i_6} \cdot c_{j_3} + c_{i_5} \cdot c_{j_2}) - b_{ij} \cdot (c_{i_6} \cdot c_{j_2} - c_{i_5} \cdot c_{j_3})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^P}{\partial \theta_i}) = g_{ij} \cdot (c_{i_5} \cdot c_{j_6} - c_{i_6} \cdot c_{j_5}) - b_{ij} \cdot (c_{i_6} \cdot c_{j_6} + c_{i_5} \cdot c_{j_5})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^P}{\partial \theta_j}) = -g_{ij} \cdot (c_{i_5} \cdot c_{j_6} - c_{i_6} \cdot c_{j_5}) + b_{ij} \cdot (c_{i_6} \cdot c_{j_6} + c_{i_5} \cdot c_{j_5})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^Q}{\partial V_i}) = -2c_{i_1} \cdot (b_{ij} + b_{ij}^s) + b_{ij} \cdot (c_{j_6} \cdot c_{i_3} + c_{j_5} \cdot c_{i_2}) - g_{ij} \cdot (c_{j_6} \cdot c_{i_2} - c_{j_5} \cdot c_{i_3})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^Q}{\partial V_j}) = b_{ij} \cdot (c_{i_6} \cdot c_{j_3} + c_{i_5} \cdot c_{j_2}) - g_{ij} \cdot (c_{i_5} \cdot c_{j_3} - c_{i_6} \cdot c_{j_2})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^Q}{\partial \theta_i}) = -b_{ij} \cdot (c_{i_5} \cdot c_{j_6} - c_{i_6} \cdot c_{j_5}) - g_{ij} \cdot (c_{i_6} \cdot c_{j_6} + c_{i_5} \cdot c_{j_5})$		
$\text{Enc}_{\text{SE}}(\frac{\partial f_{ij}^Q}{\partial \theta_j}) = b_{ij} \cdot (c_{i_5} \cdot c_{j_6} - c_{i_6} \cdot c_{j_5}) + g_{ij} \cdot (c_{i_6} \cdot c_{j_6} + c_{i_5} \cdot c_{j_5})$		

that $\mathcal{A}_1$ learns from the real protocol execution nothing more than from the ideal protocol execution which provides security and privacy. In other words, a protocol protects privacy in a semi-honest adversarial model if whatever can be computed by a party participating in the protocol can be computed based on its input and output only.

Definition 1. *The protocol π realises state estimation functionality with privacy protection against a probabilistic-polynomial time adversary $\mathcal{A}_1$ who controls SO and colluded TGCs in a semi-honest adversarial model if there exists a probabilistic -polynomial-time algorithm $\mathcal{S}$ generating simulated views for the adversary $\mathcal{A}_1$ such that:*

$$\{\mathcal{S}(\lambda, \mathcal{I}_{\mathcal{A}_1}, \mathcal{O}_{\mathcal{A}_1})\} \stackrel{\text{c}}{\equiv} \{\textit{view}_{\mathcal{A}_1}^{\pi}(\lambda, \mathcal{I})\}$$

where λ is the security parameter, $\mathcal{I}_{\mathcal{A}_1}, \mathcal{O}_{\mathcal{A}_1}$ are the input and output of the adversary $\mathcal{A}_1$, $\mathcal{I}$ is the input of all parties. $\mathcal{S}(\lambda, \mathcal{I}_{\mathcal{A}_1}, \mathcal{O}_{\mathcal{A}_1})$ is the simulated view, $\textit{view}_{\mathcal{A}_1}^{\pi}(\lambda, \mathcal{I})$ is the adversary $\mathcal{A}_1$'s real view in an execution of protocol π which includes the adversary's input, internal random tapes, and incoming messages.

In the proposed privacy-preserving protocols for DC and AC state estimation, what the adversary $\mathcal{A}_1$ can have to deduce some information about an honest party are its input, output, and incoming encrypted messages. Informally, the above formal definition implies that what the adversary $\mathcal{A}_1$ learns about the private data of an honest party TGC_h from the protocol execution is no more than what she/he can derive from her/his input and output. Obtaining incoming encrypted messages of the target honest TGC's private data in a real execution of the protocol does not add up more information for the adversary $\mathcal{A}_1$. In the followings, we will prove that both π_{DC} and π_{AC} satisfy Definition 1 in terms of providing semantic security protection of the private input and output against an adversary $\mathcal{A}_1$ controlling SO and colluded parties TGCs.

a. Privacy-preserving DC state estimation

We prove that π_{DC} realises DC state estimation functionality with privacy protection against a probabilistic-polynomial time adversary $\mathcal{A}_1$ who controls SO and colluded TGCs in a semi-honest adversarial model with regard to Definition 1. That is, there exists a probabilistic polynomial-time algorithm $\mathcal{S}_1^{\text{DC}}$ such that the generated views by $\mathcal{S}_1^{\text{DC}}$ are computationally indistinguishable from the real views of the adversary $\mathcal{A}_1$ in a real execution of protocol π_{DC}:

$$\{\mathcal{S}_1^{\text{DC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}})\} \stackrel{\text{c}}{\equiv} \{\text{view}_1^{\pi_{\text{DC}}}\} \tag{58}$$

where λ is the security parameter, $\mathcal{I}_{\mathcal{C}}$ is the input of $\mathcal{A}_1$'s corrupted parties, $\mathcal{O}_{\mathcal{C}}$ is the output of $\mathcal{A}_1$'s corrupted parties.

The view of $\mathcal{A}_1$ who controls colluded TGCs and SO during an excecution of π_{DC} consists of the inputs, the internal random tapes of corrupted parties, and all the messages corrupted parties received [39], which is:

$$\begin{aligned} \text{view}_1^{\pi_{\text{DC}}} = ((&sk_c, \boldsymbol{H}_c, \boldsymbol{y}_c)_{c \in \mathcal{C} \setminus \text{SO}}, \\ &(\boldsymbol{H}_0, \boldsymbol{y}_0, \boldsymbol{R}, \boldsymbol{r}), (pk_i, ek_i)_{i \in \mathcal{H} \cup \{\text{SE}\}}, \\ &\mathcal{E}_{i, i \in \{1 \cdots k\}}, \mathcal{E}_{\text{SE}}) \end{aligned} \tag{59}$$

where $\mathcal{E}_i$ is the set of all ciphertexts using TGC_i's public key, $\mathcal{E}_{\text{SE}}$ is the set of all ciphertexts using SE's public key sent to $\mathcal{A}_1$'s corrupted parties:

$$\mathcal{E}_{\text{SE}} = (\{\boldsymbol{EA}_i, \boldsymbol{eb}_i\}_{i \in \{0 \cdots k\}}) \tag{60}$$

$$\mathcal{E}_i = \{\boldsymbol{e\hat{x}}_i^*\}_{i \in \{1 \cdots k\}} \tag{61}$$

$\mathcal{S}_1^{\text{DC}}$ is given the security parameter λ, input $\mathcal{I}_{\mathcal{C}} = \{\boldsymbol{H}_c, \boldsymbol{y}_c\}_{c \in \mathcal{C}}$ and output $\mathcal{O}_{\mathcal{C}} = \{\hat{\boldsymbol{x}}_c\}_{c \in \mathcal{C} \setminus \{\text{SO}\}}$ of $\mathcal{A}_1$'s colluded parties, and works to generate the view for $\mathcal{A}_1$ as follows:

- $\mathcal{S}_1^{\text{DC}}$ honestly follows the protocol to generate the sets of keys (sk', pk', ek'), $\boldsymbol{R'}$, $\boldsymbol{r'}$.

- Due to the fact that $\mathcal{S}_1^{\text{DC}}$ does not have the input and output of the honest TGC$_h$, which is $(\boldsymbol{H}_h, \boldsymbol{y}_h, \hat{\boldsymbol{x}}_h)$, it sets the 'garbage' data of the $n \times n$ identity matrix and the zero-vector of n components for the honest parties' data instead.
- $\mathcal{S}_1^{\text{DC}}$ honestly follows the protocol to generate the encryption set $\mathcal{E}'_{\text{SE}}$, which is:

$$\mathcal{E}'_{\text{SE}} = (\{\boldsymbol{EA}'_c, \boldsymbol{eb}'_c\}_{c\in\mathcal{C}}, \{\text{Enc}_{\text{SE}}(\boldsymbol{I}), \text{Enc}_{\text{SE}}(\boldsymbol{0})\}_{h\in\mathcal{H}}) \tag{62}$$

- $\mathcal{S}_1^{\text{DC}}$ calculates the masked states $\hat{\boldsymbol{x}}_i^{*'}$:

$$\hat{\boldsymbol{x}}_i^{*'} = \hat{\boldsymbol{x}}_i + \boldsymbol{r}'_i \tag{63}$$

where $\hat{\boldsymbol{x}}_c$ is known by $\mathcal{S}_1^{\text{DC}}$ and $\hat{\boldsymbol{x}}_h = \boldsymbol{0}$ is the 'garbage'. Then $\mathcal{S}_1^{\text{DC}}$ encrypts all masked states:

$$\{\mathcal{E}'_i\}_{i\in\{1\cdots k\}} = (\{\boldsymbol{e}\hat{\boldsymbol{x}}_c^{*'}\}_{c\in\mathcal{C}\setminus\{\text{SO}\}}, \{\text{Enc}_{\text{h}}(\boldsymbol{r}'_h)\}_{h\in\mathcal{H}}) \tag{64}$$

- $\mathcal{S}_1^{\text{DC}}$ outputs the generated view for $\mathcal{A}_1$ as:

$$\begin{aligned}\mathcal{S}_1^{\text{DC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}}) = ((&sk'_c, \boldsymbol{H}_c, \boldsymbol{y}_c)_{c\in\mathcal{C}\setminus\text{SO}},\\ &(\boldsymbol{H}_0, \boldsymbol{y}_0, \boldsymbol{R}', \boldsymbol{r}'), (pk'_i, ek'_i)_{i\in\mathcal{H}\cup\{\text{SE}\}},\\ &\mathcal{E}'_{i,i\in\{1\cdots k\}}, \mathcal{E}'_{\text{SE}})\end{aligned} \tag{65}$$

It remains to show that the distribution of the real view and the distribution of the generated view is indistinguishable. Note that, because the estimated states $\hat{\boldsymbol{x}}_h$ (the plaintexts) of the honest party TGC$_h$ is a part of the whole estimated states $\hat{\boldsymbol{x}}$ computed from $\boldsymbol{Ax} = \boldsymbol{b}$, the adversary can have some information about $\hat{\boldsymbol{x}}_h$. This derived information can be denoted as an auxiliary information $\mathcal{L}(\hat{\boldsymbol{x}}_h|(\hat{\boldsymbol{x}}_c, \boldsymbol{H}_c, \boldsymbol{y}_c))$ of the plaintext $\hat{\boldsymbol{x}}_h$ that is leaked to the adversary from the adversary's input $(\boldsymbol{H}_c, \boldsymbol{y}_c)$ and output $\hat{\boldsymbol{x}}_c$. This is the deterministic leakage from the output of the state estimation functionality given a fixed input. This leakage is independent of the random messages (i.e. the ciphertexts) generated in the proposed protocol. Importantly, the definition of semantic security also considers an arbitrary auxiliary information function of the plaintext that may be leaked to the adversary. In state estimation, the leaked information $\mathcal{L}(\hat{\boldsymbol{x}}_h|(\hat{\boldsymbol{x}}_c, \boldsymbol{H}_c, \boldsymbol{y}_c))$ of the plaintext $\hat{\boldsymbol{x}}_h$ is auxiliary information of the plaintext $\hat{\boldsymbol{x}}_h$. Hence, the indistinguishability of the view distributions can be justified by the indistinguishability of semantic security of the underlying homomorphic scheme with auxiliary information. By the semantic security of the underlying homomorphic encryption scheme with auxiliary information, the sets of the ciphertexts in the real execution and in the simulation are computationally indistinguishable. Besides, the sets of keys and random elements are identically distributed in the real execution and in the simulation (due to a semi-honest adversarial model). Therefore, the views are computationally indistinguishable. □

b. Privacy-preserving AC state estimation

We use the modular sequential composition theorem for a semi-honest adversarial model [39], [40] to prove that π_{AC} privately computes AC state estimation functionality in the present of adversary $\mathcal{A}_1$. Note that, $\pi_{\text{AC}} = \text{INIT}\|\{\pi_t\}_{t=1}^T$, then $\text{view}_1^{\pi_{\text{AC}}} = \text{view}_1^{\text{INIT}}||\{\text{view}_1^{\pi_t}\}_{t=1}^T$

Sub-protocol INIT generates the keys (sk_i, pk_i, ek_i) used in all subsequent sub-protocols π_t. The view of $\mathcal{A}_1$ who controls colluded TGCs and SO during an execution of INIT consists of the inputs, the internal random tapes of corrupted parties, and all the messages corrupted parties received:

$$\begin{aligned}\text{view}_1^{\text{INIT}} = ((&sk_c, pk_c, ek_c, \boldsymbol{H}_c(\boldsymbol{x}_c), \boldsymbol{h}_c(\boldsymbol{x}_c))_{c\in\mathcal{C}\setminus\text{SO}},\\ &(\boldsymbol{y}_0, \boldsymbol{H}_0(\boldsymbol{x}^{(bo)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)})),\\ &(pk_h, ek_h)_{h\in\mathcal{H}}, \hat{\boldsymbol{x}}^{(0)})\end{aligned} \tag{66}$$

First, we prove that π_t realises one iteration of ACSE with privacy protection against a probabilistic-polynomial time adversary $\mathcal{A}_1$ who controls SO and colluded TGCs in a semi-honest adversarial model. That is, there exist probabilistic polynomial-time algorithms $\mathcal{S}_1^t$ such that the generated views by $\mathcal{S}_1^t$ are computationally indistinguishable from the real views of the adversary $\mathcal{A}_1$ in a real execution of protocol π_t:

$$\{\mathcal{S}_1^t(\lambda, \mathcal{I}_{\mathcal{C}}^{(t)}, \mathcal{O}_{\mathcal{C}}^{(t)})\} \overset{\text{c}}{\equiv} \{\text{view}_1^{\pi_t}\} \tag{67}$$

where λ is the security parameter, $\mathcal{I}_{\mathcal{C}}^{(t)}$ is the input of corrupted parties of $\mathcal{A}_1$, $\mathcal{O}_{\mathcal{C}}^{(t)}$ is the output of corrupted parties of $\mathcal{A}_1$.

$$\begin{aligned}\mathcal{I}_{\mathcal{C}}^{(t)} =&((sk_c, pk_c, ek_c), \boldsymbol{y}_c, \hat{\boldsymbol{x}}_c^{(t)}, \boldsymbol{H}_c(\hat{\boldsymbol{x}}_c^{(t)}), \boldsymbol{h}_c(\hat{\boldsymbol{x}}_c^{(t)}))_{c\in\mathcal{C}\setminus\{\text{SO}\}},\\ &(\boldsymbol{y}_0, \boldsymbol{H}_0(\boldsymbol{x}^{(bo)(t)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)(t)})), (pk_h, ek_h)_{h\in\mathcal{H}})\end{aligned} \tag{68}$$

$$\mathcal{O}_{\mathcal{C}}^{(t)} = (cont^{(t)}, \hat{\boldsymbol{x}}_{c,c\in\mathcal{C}}^{(t+1)}) \tag{69}$$

where $cont^{(t)} = \sum_{i=1}^k (\|\hat{\boldsymbol{x}}_i^{(t+1)} - \hat{\boldsymbol{x}}_i^{(t)}\|_\infty > \tau ? 1 : 0)$

The view of $\mathcal{A}_1$ who controls colluded TGCs and SO during an execution of π_t consists of the inputs, the internal random tapes of corrupted parties, and all the messages corrupted parties received, which is:

$$\begin{aligned}\text{view}_1^{\pi_t} = ((&sk_c, pk_c, ek_c, \boldsymbol{y}_c, \hat{\boldsymbol{x}}_c^{(t)},\\ &\boldsymbol{H}_c(\hat{\boldsymbol{x}}_c^{(t)}), \boldsymbol{h}_c(\hat{\boldsymbol{x}}_c^{(t)}))_{c\in\mathcal{C}\setminus\text{SO}},\\ &(\boldsymbol{R}^{(t)}, \boldsymbol{r}^{(t)}, \boldsymbol{y}_0, \boldsymbol{H}_0(\boldsymbol{x}^{(bo)(t)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)(t)})),\\ &(pk_h, ek_h)_{h\in\mathcal{H}\cup\{\text{SE}\}}, cont^{(t)},\\ &\mathcal{E}_{i,i\in\{1\cdots k\}}^{(t)}, \mathcal{E}_{\text{SE}}^{(t)})\end{aligned} \tag{70}$$

where $\mathcal{E}_i^{(t)}$ is the set of incoming ciphertexts using TGC$_i$'s public key, $\mathcal{E}_{\text{SE}}^{(t)}$ is the set of incoming ciphertexts using SE's public key sent to $\mathcal{A}_1$'s corrupted parties at the t-th iteration:

$$\begin{aligned}\mathcal{E}_{\text{SE}}^{(t)} =&\{\{\boldsymbol{EA}_i^{(t)}, \boldsymbol{eb}_i^{(t)}, \{\boldsymbol{c}_b^{(t)}\}_{b\in\mathcal{B}_i}, e\beta_i^{(t)}\}_{i\in\{1\cdots k\}},\\ &\boldsymbol{EA}_0^{(t)}, \boldsymbol{eb}_0^{(t)}\}\end{aligned} \tag{71}$$

$$\mathcal{E}_i^{(t)} = \{\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_i^{*(t)}\}_{i\in\{1\cdots k\}} \tag{72}$$

$\mathcal{S}_1^t$ is given the security parameter λ, input $\mathcal{I}_{\mathcal{C}}^{(t)}$, and output $\mathcal{O}_{\mathcal{C}}^{(t)}$ of the colluded parties and works to generate the view for $\mathcal{A}_1$ as follows:

- $\mathcal{S}_1^t$ honestly follows the protocol to sample $\boldsymbol{R}^{(t)}, \boldsymbol{r}^{(t)}$.
- Due to the fact that $\mathcal{S}_1^t$ does not have the input and output of the honest TGC$_h$, it sets the 'garbage' data of the $n \times n$ identity matrix, the zero-vector

of n components, and the zero value for the honest parties' data instead.

- $\mathcal{S}_1^t$ honestly follows the protocol to generate the encryption set $\mathcal{E}'_{\text{SE}}$, which is:

$$\begin{aligned}\mathcal{E}_{\text{SE}}^{(t)'} =&(\{\boldsymbol{EA}_c^{(t)'}, \boldsymbol{eb}_c^{(t)'}, \{\boldsymbol{c}_b^{(t)'}\}_{b\in\mathcal{B}_c}, e\beta_c^{(t)'}\}_{c\in\mathcal{C}\setminus\{\text{SO}\}},\\ &\{\text{Enc}_{\text{SE}}(\boldsymbol{I}), \text{Enc}_{\text{SE}}(\boldsymbol{0}), \{\text{Enc}_{\text{SE}}(\boldsymbol{0})\}_{b\in\mathcal{B}_h}, \text{Enc}_{\text{SE}}(0)\}_{h\in\mathcal{H}},\\ &\text{Enc}_{\text{SE}}(\boldsymbol{I}), \text{Enc}_{\text{SE}}(\boldsymbol{0}))\end{aligned} \tag{73}$$

- $\mathcal{S}_1^t$ calculates the masked of the state difference $\Delta\hat{\boldsymbol{x}}_i^{*(t)}$:

$$\Delta\hat{\boldsymbol{x}}_i^{*(t)} = \Delta\hat{\boldsymbol{x}}_i^{(t)} + \boldsymbol{r}_i^{(t)} \tag{74}$$

where $\Delta\hat{\boldsymbol{x}}_c^{(t)} = \hat{\boldsymbol{x}}_c^{(t+1)} - \hat{\boldsymbol{x}}_c^{(t)}$ is known by $\mathcal{S}_1^t$ and $\Delta\hat{\boldsymbol{x}}_h^{(t)} = \boldsymbol{0}$ is the 'garbage'. Then $\mathcal{S}_1^t$ encrypts all the masked of the state difference:

$$\begin{aligned}\{\mathcal{E}_i^{(t)'}\}_{i\in\{1\cdots k\}} =&(\{\boldsymbol{e}\Delta\hat{\boldsymbol{x}}_c^{*(t)'}\}_{c\in\mathcal{C}\setminus\{SO\}},\\ &\{\text{Enc}_{\text{h}}(\boldsymbol{r}_h^{(t)})\}_{h\in\mathcal{H}})\end{aligned} \tag{75}$$

- $\mathcal{S}_1^t$ outputs the generated view for $\mathcal{A}_1$ as:

$$\begin{aligned}\mathcal{S}_1^t(\lambda, \mathcal{I}_{\mathcal{C}}^{(t)}, \mathcal{O}_{\mathcal{C}}^{(t)}) = ((&sk_c, pk_c, ek_c, \boldsymbol{y}_c, \hat{\boldsymbol{x}}_c^{(t)},\\ &\boldsymbol{H}_c(\hat{\boldsymbol{x}}_c^{(t)}), \boldsymbol{h}_c(\hat{\boldsymbol{x}}_c^{(t)}))_{c\in\mathcal{C}\setminus\text{SO}},\\ &(\boldsymbol{R}^{(t}, \boldsymbol{r}^{(t)}, \boldsymbol{y}_0, \boldsymbol{H}_0(\boldsymbol{x}^{(bo)(t)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)(t)})),\\ &(pk_h, ek_h)_{h\in\mathcal{H}\cup\{\text{SE}\}}, cont^{(t)},\\ &\{\mathcal{E}^{(t)'}\}_{i,i\in\{1\cdots k\}}, \mathcal{E}_{\text{SE}}^{(t)'})\end{aligned} \tag{76}$$

From Eq. (70) (76) and the semantic security of the underlying homomorphic encryption scheme, the sets of the ciphertexts in the real execution and in the simulation are computationally indistinguishable. Besides, the sets of keys and random elements are identically distributed in the real execution and in the simulation (due to a semi-honest adversarial model). Therefore, the views are computationally indistinguishable, and (67) is proved.

Next, to prove that π_{AC} realises AC state estimation with privacy protection against a semi-honest adversary $\mathcal{A}_1$ who controls SO and colluded TGCs, we construct a probabilistic polynomial-time algorithm $\mathcal{S}_1^{\text{AC}}$ such that the generated views by $\mathcal{S}_1^{\text{AC}}$ are computationally indistinguishable from the real views of the adversary $\mathcal{A}_1$ in a real execution of protocol π_{AC}:

$$\{\mathcal{S}_1^{\text{AC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}})\} \stackrel{\text{c}}{\equiv} \{\text{view}_1^{\pi_{\text{AC}}}\} \tag{77}$$

where λ is the security parameter, $\mathcal{I}_{\mathcal{C}}$ is the input of corrupted parties of $\mathcal{A}_1$, $\mathcal{O}_{\mathcal{C}}$ is the output of corrupted parties of $\mathcal{A}_1$.

$$\begin{aligned}\mathcal{I}_{\mathcal{C}} =&\{\{\boldsymbol{y}_c, \boldsymbol{H}_c(\boldsymbol{x}_c), \boldsymbol{h}_c(\boldsymbol{x}_c)\}_{c\in\mathcal{C}},\\ &\{\boldsymbol{y}_0, \boldsymbol{H}_0(\boldsymbol{x}^{(bo)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)})\}\}\end{aligned} \tag{78}$$

$$\mathcal{O}_{\mathcal{C}} =\{\{\hat{\boldsymbol{x}}_c\}_{c\in\mathcal{C}}, sol\} \tag{79}$$

$\mathcal{S}_1^{\text{AC}}$ is given $\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}}$ and works to generate the view for $\mathcal{A}_1$ as follows:

- $\mathcal{S}_1^{\text{AC}}$ honestly follows the protocol to generate sets of keys (sk_i, pk_i, ek_i).
- Set $t = 1$, $\hat{\boldsymbol{x}}^{(1)} = \boldsymbol{x}_{\text{flat}}$, and initialize:

$$\begin{aligned}\mathcal{S}_1^{\text{INIT}} =&((sk_c, pk_c, ek_c, \boldsymbol{H}_c(\boldsymbol{x}_c), \boldsymbol{h}_c(\boldsymbol{x}_c))_{c\in\mathcal{C}\setminus\text{SO}},\\ &(\boldsymbol{y}_0, \boldsymbol{H}_0(\boldsymbol{x}^{(bo)}), \boldsymbol{h}_0(\boldsymbol{x}^{(bo)})),\\ &(pk_h, ek_h)_{h\in\mathcal{H}}, \hat{\boldsymbol{x}}^{(0)})\end{aligned} \tag{80}$$

From Eq. (66) (80), we have

$$\{\mathcal{S}_1^{\text{INIT}}\} \stackrel{\text{c}}{\equiv} \{\text{view}_1^{\text{INIT}}\} \tag{81}$$

- Set $\mathcal{V} = \mathcal{S}_1^{\text{INIT}}$
- Repeat
  - Invoke the simulator S_1^t on the input $\mathcal{I}_{\mathcal{C}}^{(t)}$ and the output $\mathcal{O}_{\mathcal{C}}^{(t)}$ of the corrupted parties.
  - Set
$$\mathcal{V} = \mathcal{V} \parallel \mathcal{S}_1^t(\lambda, \mathcal{I}_{\mathcal{C}}^{(t)}, \mathcal{O}_{\mathcal{C}}^{(t)})$$
  - Set $t = t + 1$,

  Until $(cont^{(t-1)} == 0)||(t == T_{max} + 1)$
- Set $T = t - 1$ and output $\mathcal{V}$ as the simulated view $\mathcal{S}_1^{\text{AC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}})$ that $\mathcal{S}_1^{\text{AC}}$ generates for adversaries $\mathcal{A}_1$.

Finally, we prove that $\{\mathcal{S}_1^{\text{AC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}})\} \stackrel{\text{c}}{\equiv} \{\text{view}_1^{\pi_{\text{AC}}}\}$ using the hybrid technique of modular sequential composition theorem for semi-honest adversarial models [39] given $\{\mathcal{S}_1^t(\lambda, \mathcal{I}_{\mathcal{C}}^{(t)}, \mathcal{O}_{\mathcal{C}}^{(t)})\} \stackrel{\text{c}}{\equiv} \{\text{view}_1^{\pi_t}\}$ and $\{\mathcal{S}_1^{\text{INIT}}\} \stackrel{\text{c}}{\equiv} \{\text{view}_1^{\text{INIT}}\}$.

Denote H_t as the hybrid distribution representing the view of adversary $\mathcal{A}_1$ in an execution of π_{AC}, with the exception that the view of the INIT procedure is replaced by the simulated transcripts $\mathcal{S}_1^{\text{INIT}}$ and the views of the first t invocations of $\pi_1, \cdots, \pi_t$ are replaced by the simulated transcripts $S_1^1(\lambda, \mathcal{I}_c^{(1)}, \mathcal{O}_c^{(1)}), S_1^2(\lambda, \mathcal{I}_c^{(2)}, \mathcal{O}_c^{(2)}), \cdots, S_1^t(\lambda, \mathcal{I}_c^{(t)}, \mathcal{O}_c^{(t)})$. So, $H_T = \{\mathcal{S}_1^{\text{AC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}})\}$. Besides, $\{\text{view}_1^{\pi_{\text{AC}}}\} \stackrel{\text{c}}{\equiv} H_0$ given $\{\text{view}_1^{\text{INIT}}\} \stackrel{\text{c}}{\equiv} \{\mathcal{S}_1^{\text{INIT}}\}$, and $H_t \stackrel{\text{c}}{\equiv} H_{t+1}$ given $\{\text{view}_1^{\pi_{t+1}}\} \stackrel{\text{c}}{\equiv} \{\mathcal{S}_1^{(t+1)}(\lambda, \mathcal{I}_{\mathcal{C}}^{(t+1)}, \mathcal{O}_{\mathcal{C}}^{(t+1)})\}$ $(t = 0 \cdots T-1)$. Thus, we have:

$$\{\text{view}_1^{\pi_{\text{AC}}}\} \stackrel{\text{c}}{\equiv} H_0 \stackrel{\text{c}}{\equiv} H_1 \cdots \stackrel{\text{c}}{\equiv} H_{T-1} \stackrel{\text{c}}{\equiv} H_T = \{\mathcal{S}_1^{\text{AC}}(\lambda, \mathcal{I}_{\mathcal{C}}, \mathcal{O}_{\mathcal{C}})\} \ \square$$

6.2.2 Privacy protection against an adversary $\mathcal{A}_2$

In the proposed privacy-preserving protocols for DC and AC state estimation, the additive and multiplicative masking methods provide privacy protection against $\mathcal{A}_2$. In the followings, it will be demonstrated that the proposed schemes provide local differential privacy protection of the private output and sufficient multiplicative masking protection of the private input against an adversary $\mathcal{A}_2$.

The adversary $\mathcal{A}_2$ can only obtain the perturbed $\hat{\boldsymbol{x}}^*$ and $\Delta\hat{\boldsymbol{x}}^{(t)*}$ in DC and AC state estimation, respectively. For DC state estimation, the estimated state vector $\hat{\boldsymbol{x}}$ is masked with an additive Laplace random noise vector ($\hat{\boldsymbol{x}}^* = \hat{\boldsymbol{x}} + \boldsymbol{r}$), where $\boldsymbol{r}$ is randomly sampled from the Laplace distribution $\text{Lap}(0, v, n)$, $v = \frac{2\cdot\text{MAX}}{\varepsilon}$ ($x_i \in [-\text{MAX}, \text{MAX}]$). Thus, π_{DC} provides ε-local differential privacy protection for $\hat{\boldsymbol{x}}$ with the Laplace mechanism [33], [36], [37]. For AC state estimation, a new randomness $\boldsymbol{r}^{(t)}$ is freshly sampled at each iteration t from Laplace distribution $\text{Lap}(0, v, n)$, $v = \frac{2\cdot\text{MAX}}{\varepsilon}$. After each time of running AC state estimation, $\mathcal{A}_2$ can only obtain the random perturbed $\Delta\hat{\boldsymbol{x}}^{*(t)} =$

$\Delta\hat{\boldsymbol{x}}^{(t)} + \boldsymbol{r}^{(t)}$. Assume that, having $\Delta\hat{\boldsymbol{x}}^{*(t-1)}$, the adversary $\mathcal{A}_2$ can obtain $\hat{\boldsymbol{x}}^{*(t)} = \hat{\boldsymbol{x}}^{(t-1)} + \Delta\hat{\boldsymbol{x}}^{*(t-1)}$ (e.g., with $t = 2$, $\hat{\boldsymbol{x}}^{(t-1)} = \boldsymbol{x}_{\text{flat}}$). Note that, $\hat{\boldsymbol{x}}^{(t)} = \hat{\boldsymbol{x}}^{(t-1)} + \Delta\hat{\boldsymbol{x}}^{(t-1)}$ and $\Delta\hat{\boldsymbol{x}}^{*(t-1)} = \Delta\hat{\boldsymbol{x}}^{(t-1)} + \boldsymbol{r}^{(t-1)}$; thus, $\hat{\boldsymbol{x}}^{*(t)} = \hat{\boldsymbol{x}}^{(t)} + \boldsymbol{r}^{(t-1)}$, where $\boldsymbol{r}^{(t-1)} \leftarrow \text{Lap}(0, v, n), v = \frac{2\cdot\text{MAX}}{\varepsilon}$. Therefore, π_{AC} provides ε-local differential privacy protection for $\hat{\boldsymbol{x}}^{(t)}$ with the Laplace mechanism.

By adopting the multiplicative masking for $\boldsymbol{A}$ using an invertible matrix $\boldsymbol{R}$, from the public information that the state estimation functionality is solvable or not, an adversary can deduce the singularity of $\boldsymbol{A}$. If the state estimation problem is solvable, $\boldsymbol{A}^*$ is invertible, and then $\boldsymbol{A}$ is also invertible ($\det(\boldsymbol{R}\cdot\boldsymbol{A}) = \det(\boldsymbol{R})\cdot\det(\boldsymbol{A})$). However, even knowing about the singularity of $\boldsymbol{A}$, the adversary cannot deduce the singularity of $\boldsymbol{A}_h$ (i.e., no information of $\boldsymbol{H}_h$, $\boldsymbol{y}_h$) due to the fact that $\det(\boldsymbol{A}_c + \boldsymbol{A}_h) \neq \det(\boldsymbol{A}_c) + \det(\boldsymbol{A}_h)$. Besides, $\boldsymbol{A}_c^*$ and $\boldsymbol{b}_c^*$ are not available to TGC$_c$ due to the unknown randomness $(\boldsymbol{R}, \boldsymbol{r})$. Thus, having the relation $\boldsymbol{A}^*.\boldsymbol{x}^* = \boldsymbol{b}^*$ and $(\boldsymbol{A}_c, \boldsymbol{b}_c)$ from colluded parties does not add more information about $(\boldsymbol{A}_h^*, \boldsymbol{b}_h^*)$ than from only knowing $(\boldsymbol{A}^*, \boldsymbol{b}^*)$, given $\boldsymbol{A}^* = \boldsymbol{A}_h^* + \boldsymbol{A}_c^*$, $\boldsymbol{b}^* = \boldsymbol{b}_h^* + \boldsymbol{b}_c^*$. Moreover, from the result of the statistical properties of multiplicative noise masking for confidentiality protection [41], the efficacy of noise multiplication $\boldsymbol{R}\cdot\boldsymbol{A}$ for privacy protection of $\boldsymbol{A_h}$ is estimated formally according to the variance of the noise distribution to generate the randomness $\boldsymbol{R}$. The disclosure risk assessment is put in the scenario where an adversary knows the perturbed cell total ($\boldsymbol{A}^*[i,j] = \sum_k \boldsymbol{R}[i,k]\cdot\boldsymbol{A}[k,j]$) and tries to infer about the value of a specific cell $\boldsymbol{A}[k,j]$ (not mention the value of $\boldsymbol{A}_h[k,j]$). As from [41], the sufficient privacy for practical applications to be required would be that approximate 95% error bounds for each value $\boldsymbol{A}[k,j]$ are at least p% away from its actual value. One possibility is to set $\sigma_{\boldsymbol{R}} = p/200$ [41]. Therefore, we can choose suitable noise distribution to achieve sufficient privacy protection with multiplicative masking $\boldsymbol{R}\cdot\boldsymbol{A}$. For example, with $\boldsymbol{R} \leftarrow_{\$} [-1,1]^{n\times n}$, $\sigma_{\boldsymbol{R}} = 2/\sqrt{12}$, then $p \approx 115$.

6.3 Communication and computation analysis

6.3.1 Computation cost

We estimate computation overhead in terms of the number of homomorphic computation operations in the protocols, including homomorphic encryption (#Enc), homomorphic decryption (#Dec), and homomorphic evaluation (#Eval).

TABLE 3: Computation cost of π_{DC}

	#Enc	#Dec	#Eval
TGC$_i$	2	1	0
SO	0	0	5
SE	k	2	0

The computation cost of π_{DC} is summarised in Table 3. In π_{DC} (Fig. 2), each TGC$_i$ executes 2 encryption operations and 1 decryption operation. SO executes 5 homomorphic evaluation operations. For SE, the number of encryption and decryption operations is k and 2, respectively.

TABLE 4: Computation cost of π_{AC}

	#Enc	#Dec	#Eval
TGC$_i$	$T\cdot(3+6\cdot\lvert\mathcal{B}_i\rvert)$	T	0
SO	0	0	$T\cdot 8$
SE	$T\cdot(k+2)$	$T\cdot 5$	0

The computation cost of π_{AC} is summarised in Table 4. In π_{AC} (Fig. 3), for each π_t, each TGC$_i$ executes 3 encryption operations to get $\boldsymbol{EA}_i, \boldsymbol{eb}_i, e\beta_i$, and 6 encryption operations to get $\boldsymbol{c}_b$ corresponding to each of its boundary bus b, thus counts to $T\cdot(3+6\cdot|\mathcal{B}_i|)$ encryption operations. The number of decryption operations that each TGC$_i$ computes is T. SO executes 8 homomorphic evaluation operations. For SE, the number of encryption and decryption operations is $T\cdot(k+2)$ and $T\cdot 5$, respectively.

6.3.2 Communication cost

Table 5 provides the overall communication complexity of π_{DC} and π_{AC} in terms of the number of the plaintexts and the ciphertexts sent at each step of the protocol. Denote L_p, L_c, L_k as the size of a plaintext, a ciphertext, and a pair key (pk, ek) respectively.

In π_{DC}, each TGC$_i$ sends its (pk_i, ek_i) to SE and SO, 2 ciphertexts to SO; SO sends 2 ciphertexts to SE and k ciphertexts to k TGCs; SE sends its (pk_i, ek_i) to SO and TGCs and k ciphertexts to SO. Thus the number of key messages is: $2\cdot k + k + 1 = 3\cdot k + 1$, the number of cipher messages is: $2\cdot k + 2 + k + k = 4\cdot k + 2$.

In π_{AC}, besides $4\cdot k+2$ ciphertexts as in π_{DC}, for each π_t, there are additional 2 ciphertexts $(\boldsymbol{E}\overline{\boldsymbol{H}}_0, \boldsymbol{eh}_0^*)$ sent from SO to SE and 2 ciphertexts $(\boldsymbol{EA}_0, \boldsymbol{eb}_0)$ sent from SE to SO, $6\cdot|\mathcal{B}|$ of ciphertexts corresponding $|\mathcal{B}|$ boundary buses, k ciphertexts $e\beta_i$ sent from k TGC$_i$ to SO, and 1 ciphertext $econt$ sent from SO to SE. Thus, the total number of ciphertexts transferred is $T\cdot(4\cdot k+2+2+2+6\cdot|\mathcal{B}|+k+1) = T\cdot(5\cdot k+7+6\cdot|\mathcal{B}|)$.

7 Empirical Evaluation

In this section, the proposed privacy-preserving state estimation schemes are simulated on the IEEE 14-bus system [42]. The efficiency and scalability are then analysed on bigger systems (for example, IEEE-118 bus [43] and IEEE-300 bus [44]) with the adoption of parallel matrix computation on high-performance computing infrastructure.

The IEEE 14-bus test case represents a simple approximation of the American Electric Power system as of February 1962 [42]. It has 14 buses, 5 generators, and 11 loads (Fig. 4). The IEEE 14-bus system is divided into 3 sub-systems TGCs with the statistics of the partition of the boundary (#bo) and internal (#in) buses and lines as in Table 6, in which three sub-systems are managed by TGC$_1$, TGC$_2$, TGC$_3$ and the interconnection area is handled by SO.

DC and AC load-flow calculations are performed using the open-source power system simulator Pandapower [45] to update voltage magnitudes and phase angles throughout the system. The results of the load-flow calculation represent the true states. Measurements are generated from the true states by adding device errors which are assumed Gaussian

TABLE 5: Communication cost of π_{DC} and π_{AC}

	Communication cost
π_{DC}	$L_k \cdot (3 \cdot k + 1) + L_c \cdot (4 \cdot k + 2)$
π_{AC}	$L_k \cdot (3 \cdot k + 1) + T \cdot L_c \cdot (5 \cdot k + 3 + 6 \cdot \lvert\mathcal{B}\rvert))$

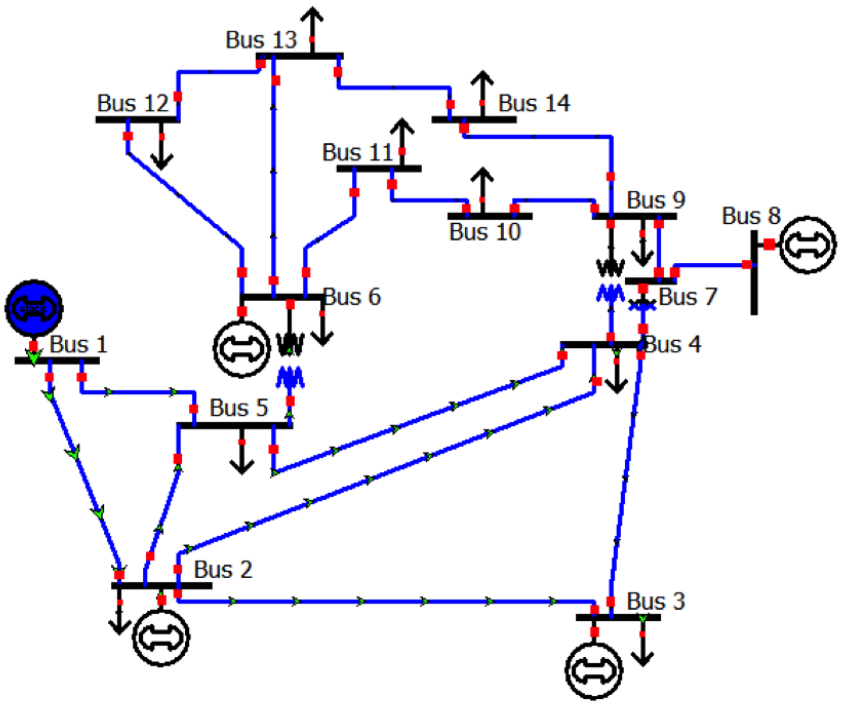

Fig. 4: IEEE 14 bus system [42]

TABLE 6: Partition of the IEEE-14 bus system

Subsystem	**bus**		**line**	
	#bo	**#in**	**#bo**	**#in**
TGC$_1$ ([1, 2, 5, 6])	3	1	0	4
TGC$_2$ ([3, 4, 7, 8, 9])	3	2	0	5
TGC$_3$ ([10, 11, 12, 13, 14])	5	0	0	3
SO	0	0	8	0

random noise with zero-mean and standard deviation of 0.01. The convergence threshold ε for AC state estimation is set to 10^{-5}. The performance metric is Root Mean Square Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} (\hat{x}_i - x_i)^2}$$

where x_i is the true state from load-flow calculation and $\hat{x}_i$ is the i-th estimated state.

The homomorphic encryption scheme CKKS [27] is adopted as the underlying homomorphic encryption. An encryption operation $\mathsf{Enc}(x)$ includes encoding x first and then encrypting. A decryption operation $\mathsf{Dec}(y)$ includes decrypting y first and then decoding. This scheme supports arithmetic operations over ciphertexts and arithmetic operations over ciphertexts and plaintexts. The scheme's security is based on the RLWE assumption over the cyclotomic ring $\mathcal{R} = \mathbb{Z}[X]/(X^\lambda + 1)$. The setting is based on [46], with $\lambda = 2^{13}$. We also utilise library HEMat, [46] which demonstrates reasonable performance for practical use (e.g. homomorphic evaluation of CNN, making a prediction based on encrypted data and model) to encrypt a matrix homomorphically and perform arithmetic evaluation on encrypted matrices.

TABLE 7: DC state estimation on IEEE-14 bus system

	DCSE	**MDCSE**	π_{DC}
RMSE of $\hat{\boldsymbol{x}}_\theta$	0.7932	0.7932	0.7928
Time (s)	0.0568	0.0972	38.689

We carry out the proposed privacy-preserving DC and AC state estimation and compare the results with the corresponding non-privacy-preserving version. As can be seen from Table 7 and Table 8, the RMSE errors of the proposed scheme are similar to the non-privacy-preserving versions. The approximation is due to the approximation property of the underlying homomorphic encryption CKKS working on real numbers. The proposed privacy-preserving state estimation does not degrade the overall state estimation accuracy significantly. For time complexity, the privacy-

TABLE 8: AC state estimation on IEEE-14 bus system

	ACSE	**MACSE**	$\pi_{\mathbf{AC}}$
RMSE of $\hat{\boldsymbol{x}}_\theta$	0.0788	0.0788	0.0793
RMSE of $\hat{\boldsymbol{x}}_V$ ($\times 10^{-3}$)	1.233	1.233	1.251
Time (s)	0.0795	0.1303	190.818

preserving versions take longer to finish than the non-privacy-preserving versions due to the homomorphic operations applied in the scheme.

Next, the efficiency and scalability of the system are analyzed on bigger systems (for example, IEEE-118 bus and IEEE-300 bus) with the adoption of parallel matrix computation on high-performance computing infrastructure.

With $\lambda = 2^{13}$, a matrix of size 64×64 can be encrypted in one ciphertext, which is sufficient for a data matrix of size 27×27 in the IEEE-14 bus system. For bigger systems like IEEE-118 bus and IEEE-300 bus, 235×235-matrices of IEEE-118 bus system and 599×599-matrices of IEEE-300 bus system are too large to be encoded into one ciphertext. The approach is to partition these large data matrices into k^2 sub-matrices, where $k = \lfloor n/64 \rfloor + 1$, and then encrypt them individually ($n = 2 \cdot N - 1$, N is the number of buses). Arithmetic operations (addition, multiplication) over large matrices can be expressed as block-wise operations over the sub-matrices of 64×64 size as the same as IEEE-14 bus system. With parallel matrix computation algorithms [47], high-performance computing can be adopted to accelerate the computation speed. For instance, DNS algorithm of matrix multiplication performs matrix multiplication in time $O(\log k) \cdot T_{64}$ by using $O(k^3/logk)$ processes where k^2 is the number of blocks and T_{64} is the time for homomorphic matrix multiplication of 64×64-matrices.

For the IEEE-118 bus system, by utilizing the computing system with $k^3/\log k = 32$ CPUs ($k = 4$) with DNS algorithms for matrix multiplication [36], the computation time can be estimated as $\log k = 2$ times as that of the IEEE-14 bus system due to the fact that homomorphic matrix multiplication consumes the most computation cost. Similarly, for the IEEE-300 bus system ($k = 10$), the corresponding computation time can be estimated as $\log k = 3.322$ times as that of the IEEE-14 bus system.

8 Conclusion

This paper designs privacy-preserving state estimation schemes for DC and AC models to solve the problem of competitive privacy in a deregulated environment of interconnected transmission grids. Private protocols based on a hybrid approach of a linear transformation for masking and a quantum-secure homomorphic encryption scheme established in the two-non-colluding-server model are designed and analysed to be secure. The proposed protocols guarantee the state estimation accuracy and the competitive privacy of each sub-grid. The results from this research motivate us to design other privacy-preserving security operations in smart grids, such as privacy-preserving false data injection detection schemes which utilize the private estimate outputs of the proposed system.

Acknowledgments

This work is partially supported by ARC Discovery Grants (DP190103660, DP200103207) and ARC Linkage Grant (LP180100663).

References

[1] J. Hu and A. V. Vasilakos, "Energy big data analytics and security: challenges and opportunities," *IEEE Transactions on Smart Grid*, vol. 7, no. 5, pp. 2423–2436, 2016.

[2] Y. Wang, Q. Chen, T. Hong, and C. Kang, "Review of smart meter data analytics: Applications, methodologies, and challenges," *IEEE Transactions on Smart Grid*, 2018.

[3] J. Shweppe and D. Rom, "Power system static state estimation: part i, ii, and iii," in *Power Industry Computer Conference*, 1969.

[4] A. Arefi, *State estimation in smart power grids*. Springer, 2011, ch. Smart Power Grids 2011.

[5] R. D. Christie, B. F. Wollenberg, and I. Wangensteen, "Transmission management in the deregulated environment," *Proceedings of the IEEE*, vol. 88, no. 2, pp. 170–195, 2000.

[6] A. Gomez-Exposito, A. Abur, A. de la Villa Jaen, and C. Gomez-Quiles, "A multilevel state estimation paradigm for smart grids," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 952–976, 2011.

[7] L. Sankar, S. Kar, R. Tandon, and H. V. Poor, "Competitive privacy in the smart grid: An information-theoretic approach," in *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2011, pp. 220–225.

[8] R. Deng, G. Xiao, R. Lu, H. Liang, and A. V. Vasilakos, "False data injection on state estimation in power systems—attacks, impacts, and defense: A survey," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 411–423, 2016.

[9] G. Hug and J. A. Giampapa, "Vulnerability assessment of ac state estimation with respect to false data injection cyber-attacks," *IEEE Transactions on Smart Grid*, vol. 3, no. 3, pp. 1362–1370, 2012.

[10] L. Sankar, "Competitive privacy: Distributed computation with privacy guarantees," in *2013 IEEE Global Conference on Signal and Information Processing*. IEEE, 2013, pp. 325–328.

[11] V. Kekatos and G. B. Giannakis, "Distributed robust power system state estimation," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1617–1626, 2012.

[12] G. N. Korres, "A distributed multiarea state estimation," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 73–84, 2010.

[13] M. Asghar, G. Dán, D. Miorandi, and I. Chlamtac, "Smart meter data privacy: A survey," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 2820–2835, 2017, cited By 3.

[14] C. Zhao, S. Zhao, M. Zhao, Z. Chen, C.-Z. Gao, H. Li, and Y.-A. Tan, "Secure multi-party computation: Theory, practice and applications," *Information Sciences*, vol. 476, pp. 357–372, 2019.

[15] P. Mukherjee and D. Wichs, "Two round multiparty computation via multi-key FHE," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2016, pp. 735–763.

[16] H. Sandberg, G. Dán, and R. Thobaben, "Differentially private state estimation in distribution networks with smart meters," in *2015 54th IEEE conference on decision and control (CDC)*. IEEE, 2015, pp. 4492–4498.

[17] Y. Kim, E. C.-H. Ngai, and M. B. Srivastava, "Cooperative state estimation for preserving privacy of user behaviors in smart grid," in *2011 IEEE international conference on smart grid communications (SmartGridComm)*. IEEE, 2011, pp. 178–183.

[18] S. Tonyali, O. Cakmak, K. Akkaya, M. M. Mahmoud, and I. Guvenc, "Secure data obfuscation scheme to enable privacy-preserving state estimation in smart grid ami networks," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 709–719, 2015.

[19] N. Kashyap, S. Werner, Y.-F. Huang, and R. Arablouei, "Privacy preserving decentralized power system state estimation with phasor measurement units," in *2016 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*. IEEE, 2016, pp. 1–5.
[20] N. Kashyap, S. Werner, and Y.-F. Huang, "Decentralized pmu-assisted power system state estimation with reduced interarea communication," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 4, pp. 607–616, 2018.
[21] J. Wang, D. Shi, J. Chen, and C.-C. Liu, "Privacy-preserving hierarchical state estimation in untrustworthy cloud environments," *IEEE Transactions on Smart Grid*, 2020.
[22] S. Kamara, P. Mohassel, and M. Raykova, "Outsourcing multiparty computation," *Cryptology ePrint Archive*, 2011.
[23] P. Mohassel and Y. Zhang, "SecureML: A System for Scalable Privacy-Preserving Machine Learning," in *2017 IEEE SYMPOSIUM ON SECURITY AND PRIVACY (SP)*, ser. IEEE Symposium on Security and Privacy, vol. 2017, no. 4. Sciendo, 2017, pp. 19–38, 38th IEEE Symposium on Security and Privacy (SP), San Jose, CA, MAY 22-26, 2017.
[24] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-Preserving Ridge Regression on Hundreds of Millions of Records," in *2013 IEEE SYMPOSIUM ON SECURITY AND PRIVACY (SP)*, ser. IEEE Symposium on Security and Privacy, 2013, pp. 334–348, 34th IEEE Symposium on Security and Privacy (SP), San Francisco, CA, MAY 19-22, 2013.
[25] I. Giacomelli, S. Jha, M. Joye, C. D. Page, and K. Yoon, "Privacy-preserving ridge regression with only linearly-homomorphic encryption," in *International Conference on Applied Cryptography and Network Security*. Springer, 2018, pp. 243–261.
[26] A. Akavia, H. Shaul, M. Weiss, and Z. Yakhini, "Linear-regression on packed encrypted data in the two-server model," in *Proceedings of the 7th ACM Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, 2019, pp. 21–32.
[27] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic Encryption for Arithmetic of Approximate Numbers," in *ADVANCES IN CRYPTOLOGY - ASIACRYPT 2017, PT I*, ser. Lecture Notes in Computer Science, Takagi, T and Peyrin, T, Ed., vol. 10624, no. I, 2017, pp. 409–437, 23rd International Conference on the Theory and Applications of Cryptology and Information Security (ASIACRYPT), Hong Kong, PEOPLES R CHINA, DEC 03-07, 2017.
[28] A. Monticelli, *State estimation in electric power systems: a generalized approach*. Springer Science & Business Media, 2012.
[29] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–487, 2013.
[30] T. Zhu, G. Li, W. Zhou, and S. Y. Philip, *Differential privacy and applications*. Springer, 2017.
[31] "Local privacy and statistical minimax rates," *Proceedings - Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pp. 429–438, 2013.
[32] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," *Advances in neural information processing systems*, vol. 27, 2014.
[33] "A comprehensive survey on local differential privacy toward data statistics and analysis," *Sensors (Switzerland)*, vol. 20, no. 24, pp. 1–48, 2020.
[34] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014, pp. 1054–1067.
[35] B. Avent, A. Korolova, S. California, D. Zeber, T. Hovden, B. Livshits, I. C. London, and B. Avent, "BLENDER : Enabling Local Search with a Hybrid Differential Privacy Model This paper is included in the Proceedings of the BLENDER : Enabling Local Search with a Hybrid Differential Privacy Model," 2017.
[36] "Guaranteeing local differential privacy on ultra-low-power systems," *Proceedings - International Symposium on Computer Architecture*, pp. 561–574, 2018.
[37] "Privacy preserving classification on local differential privacy in data centers," *Journal of Parallel and Distributed Computing*, vol. 135, pp. 70–82, 2020. [Online]. Available: https://doi.org/10.1016/j.jpdc.2019.09.009
[38] Y. Lindell, "How to simulate it–a tutorial on the simulation proof technique," *Tutorials on the Foundations of Cryptography*, pp. 277–346, 2017.
[39] O. Goldreich, *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.
[40] R. Canetti, "Security and composition of multiparty cryptographic protocols," *Journal of Cryptology*, vol. 13, no. 1, pp. 143–202, 2000.
[41] T. K. Nayak, B. Sinha, and L. Zayatz, "Statistical properties of multiplicative noise masking for confidentiality protection," *Journal of Official Statistics*, vol. 27, no. 3, p. 527, 2011.
[42] "Ieee 14-bus-system," https://icseg.iti.illinois.edu/ieee-14-bus-system/, Tech. Rep., Accessed on 21 June 2021.
[43] "Ieee 118-bus-system," https://icseg.iti.illinois.edu/ieee-118-bus-system/, Tech. Rep., Accessed on 21 December 2021.
[44] "Ieee 300-bus-system," https://icseg.iti.illinois.edu/ieee-300-bus-system/, Tech. Rep., Accessed on 21 December 2021.
[45] "Pandapower," http://www.pandapower.org/, Tech. Rep., Accessed on 21 June 2021.
[46] X. Jiang, K. Lauter, M. Kim, and Y. Song, "Secure outsourced matrix computation and application to neural networks," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 1209–1222, 2018.
[47] E. Dekel, D. Nassimi, and S. Sahni, "Parallel matrix and graph algorithms," *SIAM Journal on computing*, vol. 10, no. 4, pp. 657–675, 1981.

Hong-Yen Tran is currently a PhD student at the School of Engineering and IT, University of New South Wales, Canberra, Australia. Her research interests are in the field of information security, privacy-preserving data analytics, applied cryptography in cyber physical security.

Jiankun Hu is currently a Professor with the School of Engineering and IT, University of New South Wales, Canberra, Australia. He is also an invited expert of Australia Attorney-General's Office, assisting the draft of Australia National Identity Management Policy. He has received nine Australian Research Council (ARC) Grants and has served at the Panel on Mathematics, Information, and Computing Sciences, Australian Research Council ERA (The Excellence in Research for Australia) Evaluation Committee 2012. His research interests are in the field of cyber security covering intrusion detection, sensor key management, and biometrics authentication. He has many publications in top venues, including the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTION COMPUTERS, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, Pattern Recognition, and the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS. He is a senior area editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY.

Hemanshu R. Pota received B.E. from Sardar Vallabhbhai Regional College of Engineering and Technology, Surat, India, in 1979, M.E. from the Indian Institute of Science, Bangalore, India, in 1981, and the Ph.D. from the University of Newcastle, NSW, Australia, in 1985; all in Electrical Engineering. He is currently an associate professor at the University of New South Wales, Canberra, Australia. He has held visiting appointments at the Columbia University, New York City, NY; University of California, Los Angeles; the University of Delaware; Iowa State University; Kansas State University; Old Dominion University; the University of California, San Diego; and Centre for AI and Robotics, Bangalore.