5-2023

Modeling and Visualization of Competing Escalation Dynamics: A Multilayer Multiagent Network Approach

Josh Allen
University of Nebraska-Lincoln, josh.allen@huskers.unl.edu

MODELING AND VISUALIZATION OF COMPETING ESCALATION DYNAMICS: A MULTILAYER MULTIAGENT NETWORK APPROACH

by

Josh Allen

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Master of Science

Major: Electrical Engineering

Under the Supervision of Professor Qing Hui

Lincoln, Nebraska

May, 2023

MODELING AND VISUALIZATION OF COMPETING ESCALATION DYNAMICS: A MULTILAYER MULTIAGENT NETWORK APPROACH

Josh Allen, M.S.

University of Nebraska, 2023

Adviser: Qing Hui

Recent advances in military technology, such as hypersonic missiles, which can travel at more than five times the speed of sound and descend quickly into the atmosphere, give world nuclear superpowers a new edge. These advances up the game for nuclear superpowers with an extremely rapid, intense burst of military striking capability to secure upfront gains before encountering potentially overwhelming military confrontation. However, this so-called fait accompli has not been systematically studied by the United States in the perspective of the escalation philosophies of nuclear power competitors, or the mathematical modeling and visualization of multi-modal escalation dynamics. This gap may hamper any further command and control for nuclear deployment and decision making for strategic planning in preparation of such scenarios. This thesis aims to bridge the gap by implementing a network approach to model the escalation dynamics among competing nuclear superpowers.

ACKNOWLEDGMENTS

Special thanks to to my advisor, Dr. Qing Hui, for his expertise and guidance. I'd also like to acknowledge all others who have assisted me on my way.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

To begin modeling the escalation dynamics among several players, the first step is to create a reasonable scheme that can be used to describe spatial-temporal evolution and visualization of escalation dynamics. In this project, we target the multiagent method [3] and algebraic graph theory [12] based on the advisor's prior research [14]. This is set up as an interconnected, graphical representation by viewing each important variable in escalation dynamics as an agent, and combining them together with a networked structure to characterize agents' interplay. However, this combined method alone does not specify the microscopic dynamics underlying the graphical representation, which is crucial for quantitative analysis and visualization of escalation. To solve this issue, game theory and adversarial learning techniques are used to establish the intrinsic underlying dynamics.

Non-cooperative game theory [19] is a powerful tool to describe the competing dynamics of multiple agents in the absence of collaboration or communication from any of the others. In a real-world situation, the arm race among nuclear superpowers depicts a much more complicated picture than just non-cooperative games. For instance, it is known that the competing escalation dynamics can be predominantly described by non-cooperative games. There are certain restrictions in reality to force nuclear superpowers to compromise at some degree, e.g., availability of en-

riched uranium, sanctions, and arm treaties. Hence, a hybrid game, which consists of non-cooperative gaming as the primary dynamic, and cooperative gaming as the secondary dynamic, would be better suited to capture the multi-modal nature of escalation dynamics. This leads to a multilayer interconnected complex system. Nevertheless, how to synergistically integrate non-cooperative games with cooperative ones to accurately model intrinsic dynamics of competing escalation remains an open research area.

One of the intriguing advancements in artificial intelligence in recent years is the development of generative adversarial networks (GANs) [13] in which two deep network models are competing against each other to learn how to improve their prediction. This interesting idea sheds some light on the development of a possible mathematical tools to connect some hidden, conflicting, and correlated variables of escalation dynamics that cannot be usually represented by means of the existing physical, statistical, economic, or sociological theories. It presents a possible framework to model complex micro dynamics with certain degree of confidence. However, the downside of this method is its heavy reliance on large data training sets. When modeling competing dynamics of nuclear superpowers, such datasets are not always available. One way of circumventing this difficulty is to use Bayesian learning to update network models via sparse data. This Bayesian learning concept will lead to a separate, parallel task of multi-cue multi-choice (MCMC) decision making [10] to govern the process of network model adaptation.

With the proposed idea of hybrid games and the concept of Bayesian learning, the detailed microscopic dynamics underlying the multiagent network may be modeled. However, the real challenge is to put these ideas into work by having a tractable, computational way to predict what will happen in escalation dynamics. Moreover, what matters most for applications is to visualize the outcomes for a decision maker to

better understand the context so they may fairly assess the situation. In this project, a detailed realization of the above ideas under a multilayer multiagent network, coupled with diffusion-based propagation dynamics to model complex interdependency dynamics of the competing escalation, is presented. Specifically, the upper layer of the network model represents information gathering, data fusion, and data mining dynamics characterized by interplay between the information retrieving network (RN), information analyzing network (AN), and information formulating network (FN). The networks are modeled by an input-output cellular network, a Bayesian learning network, and an artificial neural network, respectively.

The top layer includes RN, AN, and FN. The lower layer is comprised of the hybrid game model associated with factors resulting from the top layer. This two-layer structure models alternative interdependencies among different agents in the lower layer: cooperative (pro) interaction and competing (con) interaction. This hybrid game utilizes a version of the mixed multiagent non-cooperative game and potential-based cooperative game [5] to decide the likely outcomes. From the multiagent perspective, this corresponds to two sides of impact from its input: cooperative component and compromise component. The cooperative component, reflects the steady, coordinated nuclear deployment strategy in the escalation and is the result of the potential-based cooperative game modeled by a compartmental network. It is based on the advisor's prior work [14, 22]. The compromise component, depicting negative contribution due to adversarial effects and competitive dynamics in the escalation, is the result of inhibitory effect in a network model. Finally, the overall outcome of the hybrid game will be the results from both games weighted by their priority.

1.1 Contributions

The work done to create this thesis can be categorized into three different parts: built upon the advisor's previous work, new contributions from advisor, and new contributions from myself. My advisor's previous work in swarm optimization and multiagent coordination was the basis of the optimization algorithm that was implemented throughout this project. My advisor began by creating the initial mathematical framework for each of the networks outlined in the thesis paper. I was able to assist my advisor in iterating through each of the mathematical frameworks and refactoring them to create concise and correct mathematical formulations. This process allowed me to gain a better of understanding of mathematical modeling and graphical analysis, while contributing to the thesis. Once the mathematical framework was completed I created software packages for the networks outlined in this thesis. Taking the models from the mathematical formulations that can be seen in this paper and coding them into MATLAB packages was my main contribution to this project. In addition to these contributions, I also assisted in presenting the model to NSRI.

Chapter 2

Modeling of the positive correlation between relevant factors

Motivated by the intrinsic connection between cooperative control and potential games [16], a cooperative game model is developed to represent the positive correlation between relevant factors on internal nuclear deployment within each nuclear superpower. This cooperative game model then serves as a foundation to describe the non-conflict dynamics among several nuclear superpowers.

To address the dynamic correlation among relevant variables in nuclear escalation, a dynamic model is proposed for characterizing the transient correlation and steady-state correlation between different factors.

Modeling the static correlation within one nuclear superpower is outlined as follows. Consider the following example of a structural equation model (SEM) to represent a static relationship between five variables x_i with five external inputs u_i,

$j = 1, 2, \ldots, 5$:

$$x_1 = u_1, \tag{2.1}$$

$$x_2 = a \cdot x_1 + u_2, \tag{2.2}$$

$$x_3 = b \cdot x_2 + u_3, \tag{2.3}$$

$$x_4 = e \cdot x_1 + d \cdot x_3 + u_4, \tag{2.4}$$

$$x_5 = c \cdot x_3 + f \cdot x_4 + u_5 \tag{2.5}$$

where $a, b, c, d, e, f \in \mathbb{R}$ are constant coefficients. Note that the SEM (2.1)–(2.5) can be rewritten as the following matrix form

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ a & 0 & 0 & 0 & 0 \\ 0 & b & 0 & 0 & 0 \\ e & 0 & d & 0 & 0 \\ 0 & 0 & c & f & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix}$$

In general, suppose that the causal graph for our problem is given by a direct graph (or digraph) $G = (V, E)$, where $V = \{1, \ldots, n\}$ is the index set of nodes or vertexes representing the relevant variables and $E = \{(i, j) : i \in V, j \in V, i \neq j\}$ is the set of ordered pairs (i, j) denoting the directional edge from node j to node i in the digraph G. Then the corresponding SEM can be written as

$$x_i = \sum_{(i,j) \in E} a_{ij} \cdot x_j + u_i, \quad i = 1, \ldots, n \tag{2.6}$$

where $a_{ij} \in \mathbb{R}$, $i, j = 1, \ldots, n$. To specify the detailed structure of the SEM, one needs to elaborate both the topology G and the value of a_{ij}, due to the fact that the

graph information G is implicitly embedded in the model (2.6) alongside a_{ij}. One way of representing a causal graph G for the SEM is to use the adjacency matrix $C = [c_{pq}]_{p,q=1,\ldots,n} \in \mathbb{R}^{n\times n}$ defined by

$$c_{pq} = \begin{cases} 1, & (p,q) \in E \\ 0, & (p,q) \notin E \end{cases}$$

It follows from the definition of C that $c_{pq} \equiv 0$ for $p = q$, i.e., the adjacency matrix C is an off-diagonal matrix, where all of its diagonal elements are always zero. Using the notion of adjacency matrices, one can rewrite (2.6) as

$$x_i = \sum_{j=1}^{n} a_{ij} \cdot c_{ij} \cdot x_j + u_i, \quad i = 1, \ldots, n \tag{2.7}$$

which turns the implicit embedding of G into the explicit inclusion of G in the model.

The matrix form of (2.7) can be written as

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}c_{11} & a_{12}c_{12} & a_{13}c_{13} & \cdots & a_{1n}c_{1n} \\ a_{21}c_{21} & a_{22}c_{22} & a_{23}c_{23} & \cdots & a_{2n}c_{2n} \\ a_{31}c_{31} & a_{32}c_{32} & a_{33}c_{33} & \ldots & a_{3n}c_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}c_{n1} & a_{n2}c_{n2} & a_{n3}c_{n3} & \cdots & a_{nn}c_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_n \end{bmatrix}$$

or equivalently,

$$x = (A \circ C)x + u \tag{2.8}$$

where $x = [x_1, x_2, \ldots, x_n]^{\mathrm{T}} \in \mathbb{R}^n$, "$\circ$" denotes the Hadamard product (also known as elementwise product or Schur product), $A = [a_{ij}]_{i,j=1,\ldots,n} \in \mathbb{R}^{n\times n}$, and $u =$

$[u_1, u_2, \ldots, u_n]^{\mathrm{T}} \in \mathbb{R}^n$. Note that the diagonal elements of $A \circ C$ are always zero due to the fact that $c_{ii} \equiv 0$ for all $i = 1, \ldots, n$.

The problem with the form of (2.8) is that it is unclear which variable is the input signal and which variable is the output signal, since the left-hand side and right-hand side both have x. To make it clear from the system-theoretic perspective, we rewrite (2.8) as the following input-output form

$$z = Wx + u \tag{2.9}$$

$$x = z \tag{2.10}$$

$$y = x \tag{2.11}$$

where $z = [z_1, z_2, \ldots, z_n]^{\mathrm{T}}$ denotes the intermediate signal to calculate the state vector and $y = [y_1, y_2, \ldots, y_n]^{\mathrm{T}}$ denotes the system output signal that the other agents can observe, $W = (A \circ C) \in \mathbb{R}^{n \times n}(G)$ is a graph-structured off-diagonal matrix given by the form

$$W = \begin{bmatrix} 0 & a_{12}c_{12} & a_{13}c_{13} & \cdots & a_{1n}c_{1n} \\ a_{21}c_{21} & 0 & a_{23}c_{23} & \cdots & a_{2n}c_{2n} \\ a_{31}c_{31} & a_{32}c_{32} & 0 & \ldots & a_{3n}c_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}c_{n1} & a_{n2}c_{n2} & a_{n3}c_{n3} & \cdots & 0 \end{bmatrix}$$

$\mathbb{R}^{n \times n}(G)$ denotes the set of all off-diagonal matrices with the structural topology described by G. The left-hand side of (2.9)–(2.11) represents the resulted signals while the right-hand side of (2.9)–(2.11) represents the input signals.

In this task, W is estimated under a given graph topology C by using the information of N measurements from the output signal y, intermediate signal z, state

vector x, and input signal u. In this case, W needs to be viewed as the variable ($n^2 - n$ unknown coefficients a_{ij}) in (2.9). However, the form of (2.9) does not clearly show this representation for W as x is presented as the variable and W is presented as the coefficient matrix. To change this formulation, we denote the N measurements for y, z, x, u by $y[l] = [y_{l1}, y_{l2}, \ldots, y_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$, $z[l] = [z_{l1}, z_{l2}, \ldots, z_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$, $x[l] = [x_{l1}, x_{l2}, \ldots, x_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$, and $u[l] = [u_{l1}, u_{l2}, \ldots, u_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$, respectively, $l = 1, \ldots, N$. Define the measurement matrices Y, Z, X, U as follows

$$Y = \begin{bmatrix} y[1] & y[2] & \ldots & y[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{N1} & y_{N2} & \cdots & y_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$Z = \begin{bmatrix} z[1] & z[2] & \ldots & z[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1n} \\ z_{21} & z_{22} & \cdots & z_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & \cdots & z_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$X = \begin{bmatrix} x[1] & x[2] & \ldots & x[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$U = \begin{bmatrix} u[1] & u[2] & \ldots & u[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ u_{21} & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N1} & u_{N2} & \cdots & u_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

Under these notations, (2.9) can be written as

$$
\begin{aligned}
&z[l] = Wx[l] + u[l], \quad l = 1, \ldots, N \\
&\begin{bmatrix} z[1] & z[2] & \ldots & z[N] \end{bmatrix} = W \begin{bmatrix} x[1] & x[2] & \ldots & x[N] \end{bmatrix} + \begin{bmatrix} u[1] & u[2] & \ldots & u[N] \end{bmatrix} \\
&\begin{bmatrix} z[1] & z[2] & \ldots & z[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} x[1] & x[2] & \ldots & x[N] \end{bmatrix}^{\mathrm{T}} W^{\mathrm{T}} + \begin{bmatrix} u[1] & u[2] & \ldots & u[N] \end{bmatrix}^{\mathrm{T}}
\end{aligned}
$$

or equivalently,

$$Z^{\mathrm{T}} = WX^{\mathrm{T}} + U^{\mathrm{T}} \tag{2.12}$$

$$Z = XW^{\mathrm{T}} + U \tag{2.13}$$

$$X = Z \tag{2.14}$$

$$Y = X \tag{2.15}$$

Hence, (2.12) or (2.13) shows a fundamental relationship between the unknown matrix W and the measurement matrices Z, X, Y, U.

To determine whether (2.12) has a solution W for given Z, X, Y, U, the "vec" operator is introduced for a matrix $A \in \mathbb{R}^{n \times m}$ as the vector of dimension $nm \times 1$ by stacking the columns of A vertically. For example, if

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$$

then

$$\operatorname{vec} A = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{12} \\ a_{22} \\ a_{13} \\ a_{23} \end{bmatrix}$$

Recall that for $A \in \mathbb{R}^{n\times m}$, $X \in \mathbb{R}^{m\times p}$, and $B \in \mathbb{R}^{p\times q}$,

$$\operatorname{vec}(AXB) = (B^{\mathrm{T}} \otimes A)\operatorname{vec} X$$

where $\otimes$ denotes the Kronecker product. Taking the vec operation on both sides of (2.12) yields

$$\operatorname{vec} Z^{\mathrm{T}} = \operatorname{vec}(WX^{\mathrm{T}}) + \operatorname{vec} U^{\mathrm{T}}$$

and hence,

$$\operatorname{vec} Z^{\mathrm{T}} = (X \otimes I_n)\operatorname{vec} W + \operatorname{vec} U^{\mathrm{T}}$$

or equivalently,

$$\operatorname{vec} Z^{\mathrm{T}} = (X \otimes I_n)\operatorname{vec}(C \circ A) + \operatorname{vec} U^{\mathrm{T}} \tag{2.16}$$

Note that some elements in $\operatorname{vec} W \in \mathbb{R}^{n^2\times 1}$ are always zero (diagonal ones in W). Hence, $\operatorname{vec} W$ has $n^2 - n$ free variables.

The following result is a well-known one in linear algebra.

Lemma 1. *Let $A \in \mathbb{R}^{n\times m}$ and $b \in \mathbb{R}^n$. Then the following statements are equivalent:*

i) *The equation $Ax = b$ has at least one solution $x \in \mathbb{R}^m$.*

ii) $\operatorname{rank} A = \operatorname{rank} \begin{bmatrix} A & b \end{bmatrix}$.

iii) *There exists $y \in \mathbb{R}^m$ such that $x = A^\dagger b + (I_m - A^\dagger A)y$, where $A^\dagger$ denotes the Penrose-Moore inverse of A.*

It follows from the above lemmas that (2.16) has at least one solution S in terms of $\operatorname{vec}(C \circ A)$ if and only if

$$\operatorname{rank}(X \otimes I_n) = \operatorname{rank} \begin{bmatrix} (X \otimes I_n) & \operatorname{vec} Z^{\mathrm{T}} - \operatorname{vec} U^{\mathrm{T}} \end{bmatrix}$$

Moreover, to find the value of A, this solution S should be compatible with the given graph topology G. Such that the equation

$$C \circ A = \operatorname{vec}^{-1} S$$

should have a solution in terms of A for a given C, where "vec^{-1}" denotes the inverse operation of "vec". Hence, in this thesis we assume these conditions hold for (2.12) or (2.13) under given Z, X, Y, U.

Chapter 3

Identifying the SEM: A least squares minimization approach

The key to using (2.9)–(2.11) for correlation modeling is to find the coefficient matrix W in the model through the input, output, and interlinked data. We assume that the observation for the output signal y in (2.11) contains some additive uncertainty, that is, let $\tilde{y}_i$ denote the observation of the output signal y_i, then $\tilde{y}_i = y_i + \Delta y_i$, where Δy_i denotes some observation uncertainty due to environmental noise or disturbance for the ith node in the graph G. Next, it follows from (2.6) that the ith node updates its value x_i based on the received information, x_j, from the jth node. However, such x_j may contain some other type of uncertainty existing in its communication channels. Let $\tilde{x}_j$ denote the received information from the jth node. Then $\tilde{x}_j = x_j + \Delta x_j$, where Δx_j denotes some information transmission uncertainty from the jth node to the ith node. We assume that the observation and information transmission among individual nodes are independent from each other, i.e., Δy_i and Δy_j are independent, and Δx_i and Δx_j are independent, $i \neq j$. In this case, the model (2.9)–(2.11) with

these external impacts can be written as

$$\tilde{z} = W(x + \Delta x) + u \tag{3.1}$$

$$x = \tilde{z} \tag{3.2}$$

$$y = x \tag{3.3}$$

$$\tilde{y} = y + \Delta y \tag{3.4}$$

where $\Delta x = [\Delta x_1, \ldots, \Delta x_n]^{\mathrm{T}}$, $\Delta y = [\Delta y_1, \ldots, \Delta y_n]^{\mathrm{T}}$, $\tilde{z} = [\tilde{z}_1, \ldots, \tilde{z}_n]^{\mathrm{T}}$, and $\tilde{y} = [\tilde{y}_1, \ldots, \tilde{y}_n]^{\mathrm{T}}$.

Note that it follows from (2.9) and (3.1)–(3.4) that

$$\tilde{x} = x + \Delta x = \tilde{z} + \Delta x = W(x + \Delta x) + u + \Delta x = W\tilde{x} + u + \Delta x$$

$$\tilde{y} = y + \Delta y = x + \Delta y = \tilde{z} + \Delta y = W(x + \Delta x) + u + \Delta y = W\tilde{x} + u + \Delta y$$

Then we have

$$\begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} = \begin{bmatrix} I_n \\ I_n \end{bmatrix} \tilde{z} + \begin{bmatrix} I_n & 0 \\ 0 & I_n \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{3.5}$$

As we mentioned before, in general Δx and Δy are different because Δx is due to communication uncertainty while Δy is due to observation uncertainty. When $\Delta x = 0$ and $\Delta y = 0$, it follows from (3.5) that $\tilde{x} = \tilde{z}$ and $\tilde{y} = \tilde{z}$. Hence, given N measurements $\tilde{x}[l] = [\tilde{x}_{l1}, \tilde{x}_{l2}, \ldots, \tilde{x}_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$, $\tilde{y}[l] = [\tilde{y}_{l1}, \tilde{y}_{l2}, \ldots, \tilde{y}_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$, and $u[l] = [u_{l1}, u_{l2}, \ldots, u_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$ of $\tilde{x}$, $\tilde{y}$, a possible choice of estimating a_{ij} is to minimize both $\|\tilde{x} - \tilde{z}\|$ and $\|\tilde{y} - \tilde{z}\|$ under some norm $\|\cdot\|$. Depending on the norm there could be a different cost function for estimating a_{ij}. In this project, three most common choices are considered: 1-norm, 2-norm, and infinity-norm.

Let the matrices $\tilde{Y}, \tilde{X}, \tilde{Z}, U$ be

$$\tilde{Y} = \begin{bmatrix} \tilde{y}[1] & \tilde{y}[2] & \dots & \tilde{y}[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \tilde{y}_{11} & \tilde{y}_{12} & \cdots & \tilde{y}_{1n} \\ \tilde{y}_{21} & \tilde{y}_{22} & \cdots & \tilde{y}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{y}_{N1} & \tilde{y}_{N2} & \cdots & \tilde{y}_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$\tilde{X} = \begin{bmatrix} \tilde{x}[1] & \tilde{x}[2] & \dots & \tilde{x}[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \tilde{x}_{11} & \tilde{x}_{12} & \cdots & \tilde{x}_{1n} \\ \tilde{x}_{21} & \tilde{x}_{22} & \cdots & \tilde{x}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{x}_{N1} & \tilde{x}_{N2} & \cdots & \tilde{x}_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$\tilde{Z} = \begin{bmatrix} \tilde{z}[1] & \tilde{z}[2] & \dots & \tilde{z}[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \tilde{z}_{11} & \tilde{z}_{12} & \cdots & \tilde{z}_{1n} \\ \tilde{z}_{21} & \tilde{z}_{22} & \cdots & \tilde{z}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{z}_{N1} & \tilde{z}_{N2} & \cdots & \tilde{z}_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

$$U = \begin{bmatrix} u[1] & u[2] & \dots & u[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ u_{21} & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N1} & u_{N2} & \cdots & u_{Nn} \end{bmatrix} \in \mathbb{R}^{N \times n}$$

where $\tilde{z}[l] = [\tilde{z}_{l1}, \tilde{z}_{l2}, \dots, \tilde{z}_{ln}]^{\mathrm{T}} \in \mathbb{R}^n$. Note that it follows from (3.1) that

$$\tilde{z}[l] = W\tilde{x}[l] + u[l]$$

or componentwise

$$\tilde{z}_{li} = \sum_{(i,j)\in E} a_{ij}\tilde{x}_{lj} + u_{li}, \quad i = 1, \dots, n, \quad l = 1, \dots, N \tag{3.6}$$

Equation (3.6) has the following matrix form

$$\tilde{Z} = \tilde{X}W^{\mathrm{T}} + U$$
$$\tilde{Z}^{\mathrm{T}} = W\tilde{X}^{\mathrm{T}} + U^{\mathrm{T}}$$

or

$$\mathrm{vec}\,\tilde{Z}^{\mathrm{T}} = (\tilde{X} \otimes I_n)\mathrm{vec}\,W + \mathrm{vec}\,U^{\mathrm{T}}$$
$$\mathrm{vec}\,\tilde{Z}^{\mathrm{T}} = ((C \circ \tilde{X}) \otimes I_n)\mathrm{vec}\,A + \mathrm{vec}\,U^{\mathrm{T}}$$

The 1-norm choice is to consider a cost function with the weighted 1-norm given by the form

$$\begin{aligned} J_1(\tilde{X}, \tilde{Y}, U, W) &= \sum_{l=1}^{N}\sum_{i=1}^{n} \lambda_i |\tilde{x}_{li} - \tilde{z}_{li}| + \sum_{l=1}^{N}\sum_{i=1}^{n} \theta_i |\tilde{y}_{li} - \tilde{z}_{li}| \\ &= \sum_{l=1}^{N}\sum_{i=1}^{n} \lambda_i \left| \tilde{x}_{li} - \left(\sum_{(i,j)\in E} a_{ij}\tilde{x}_{lj} + u_{li} \right) \right| + \sum_{l=1}^{N}\sum_{i=1}^{n} \theta_i \left| \tilde{y}_{li} \right. \\ &\quad \left. - \left(\sum_{(i,j)\in E} a_{ij}\tilde{x}_{lj} + u_{li} \right) \right| \end{aligned}$$

where $\lambda_i, \theta_i > 0$ are the given weights. Then the proposed optimization problem can be written as

$$\min_{W \in \mathbb{R}^{n\times n}(G)} J_1(\tilde{X}, \tilde{Y}, U, W)$$

and its optimal solution can be denoted by

$$W^* = \arg \min_{W \in \mathbb{R}^{n\times n}(G)} J_1(\tilde{X}, \tilde{Y}, U, W)$$

where "arg" refers to the argument of a function, i.e., $x^* = \arg\min_x f(x)$ means that $f(x^*) = \min_x f(x)$.

Lemma 2. *For a square matrix M, let* trace M *denote the sum of all diagonal entries for M. Then for any matrix $A \in \mathbb{R}^{m\times n}$,* $\mathrm{trace}(A^{\mathrm{T}}A) = \mathrm{trace}(AA^{\mathrm{T}}) = \|\mathrm{vec}\, A\|_2^2 = \|A\|_{\mathrm{F}}^2$*, where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm.*

Proof: Let $A = [a_{ij}]_{i=1,\ldots,m,j=1,\ldots,n}$. We use a fact for the Frobenius Norm $\|\cdot\|_{\mathrm{F}}$ that $\|A\|_{\mathrm{F}}^2 = \mathrm{trace}(AA^{\mathrm{T}}) = \sum_{i=1}^m \sum_{j=1}^n a_{ij}^2 = (\mathrm{vec}\, A)^{\mathrm{T}}(\mathrm{vec}\, A) = \|\mathrm{vec}\, A\|_2^2$, and a fact for the trace operation that $\mathrm{trace}(AB) = \mathrm{trace}(BA)$ for compatible A and B. ■

The 2-norm choice is to consider a cost function with the weighted 2-norm given by the form

$$
\begin{aligned}
J_2(\tilde{X},\tilde{Y},U,W) &= \sum_{l=1}^{N}\sum_{i=1}^{n}\lambda_i(\tilde{x}_{li}-\tilde{z}_{li})^2 + \sum_{l=1}^{N}\sum_{i=1}^{n}\theta_i(\tilde{y}_{li}-\tilde{z}_{li})^2 \\
&= \sum_{l=1}^{N}(\tilde{x}[l]-\tilde{z}[l])^{\mathrm{T}}\Lambda(\tilde{x}[l]-\tilde{z}[l]) + \sum_{l=1}^{N}(\tilde{y}[l]-\tilde{z}[l])^{\mathrm{T}}\Theta(\tilde{y}[l]-\tilde{z}[l]) \\
&= (\mathrm{vec}\,\tilde{X}^{\mathrm{T}} - \mathrm{vec}\,\tilde{Z}^{\mathrm{T}})^{\mathrm{T}}(I_N\otimes\Lambda)(\mathrm{vec}\,\tilde{X}^{\mathrm{T}} - \mathrm{vec}\,\tilde{Z}^{\mathrm{T}}) \\
&\quad + (\mathrm{vec}\,\tilde{Y}^{\mathrm{T}} - \mathrm{vec}\,\tilde{Z}^{\mathrm{T}})^{\mathrm{T}}(I_N\otimes\Theta)(\mathrm{vec}\,\tilde{Y}^{\mathrm{T}} - \mathrm{vec}\,\tilde{Z}^{\mathrm{T}}) \\
&= (\mathrm{vec}\,(\tilde{X}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}}))^{\mathrm{T}}(I_N\otimes\Lambda^{1/2})^{\mathrm{T}}(I_N\otimes\Lambda^{1/2})(\mathrm{vec}\,(\tilde{X}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}})) \\
&\quad + (\mathrm{vec}\,(\tilde{Y}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}}))^{\mathrm{T}}(I_N\otimes\Theta^{1/2})^{\mathrm{T}}(I_N\otimes\Theta^{1/2})(\mathrm{vec}\,(\tilde{Y}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}})) \\
&= (\mathrm{vec}\,(\Lambda^{1/2}(\tilde{X}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}})))^{\mathrm{T}}\mathrm{vec}\,(\Lambda^{1/2}(\tilde{X}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}})) \\
&\quad + (\mathrm{vec}\,(\Theta^{1/2}(\tilde{Y}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}})))^{\mathrm{T}}\mathrm{vec}\,(\Theta^{1/2}(\tilde{Y}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}})) \\
&= \|\mathrm{vec}\,(\Lambda^{1/2}(\tilde{X}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}}))\|_2^2 + \|\mathrm{vec}\,(\Theta^{1/2}(\tilde{Y}^{\mathrm{T}} - \tilde{Z}^{\mathrm{T}}))\|_2^2 \\
&= \mathrm{trace}((\tilde{X}-\tilde{Z})\Lambda(\tilde{X}-\tilde{Z})^{\mathrm{T}}) + \mathrm{trace}((\tilde{Y}-\tilde{Z})\Theta(\tilde{Y}-\tilde{Z})^{\mathrm{T}})
\end{aligned}
$$

where $\Lambda = \mathrm{diag}(\lambda_1,\lambda_2,\cdots,\lambda_n) \in \mathbb{R}^{n\times n}$ and $\Theta = \mathrm{diag}(\theta_1,\theta_2,\cdots,\theta_n) \in \mathbb{R}^{n\times n}$ are both

diagonal matrices, $\Lambda^{1/2} = \text{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \cdots, \sqrt{\lambda_n})$, $\Theta^{1/2} = \text{diag}(\sqrt{\theta_1}, \sqrt{\theta_2}, \cdots, \sqrt{\theta_n})$. Using Lemma 2 and the fact that $(B^{\mathrm{T}} \otimes A)\text{vec}(X) = \text{vec}(AXB)$. By (3.6), J_2 has the following equivalent form

$$\begin{aligned} J_2(\tilde{X}, \tilde{Y}, U, W) &= \text{trace}((\tilde{X} - \tilde{Z})\Lambda(\tilde{X} - \tilde{Z})^{\mathrm{T}}) + \text{trace}((\tilde{Y} - \tilde{Z})\Theta(\tilde{Y} - \tilde{Z})^{\mathrm{T}}) \\ &= \text{trace}((\tilde{X} - \tilde{X}W^{\mathrm{T}} - U)\Lambda(\tilde{X} - \tilde{X}W^{\mathrm{T}} - U)^{\mathrm{T}}) \\ &+ \text{trace}((\tilde{Y} - \tilde{X}W^{\mathrm{T}} - U)\Theta(\tilde{Y} - \tilde{X}W^{\mathrm{T}} - U)^{\mathrm{T}}) \end{aligned}$$

which shows that J_2 is indeed a function of $\tilde{X}$, $\tilde{Y}$, U, and W.

The proposed optimization problem is then a least squares minimization (LSM) problem given by

$$\min_{W \in \mathbb{R}^{n \times n}(G)} J_2(\tilde{X}, \tilde{Y}, U, W)$$

and its optimal solution can be denoted by

$$W^* = \arg \min_{W \in \mathbb{R}^{n \times n}(G)} J_2(\tilde{X}, \tilde{Y}, U, W)$$

The infinity-norm choice is to consider a cost function with the weighted infinity-norm given by the form

$$\begin{aligned} J_\infty(\tilde{X}, \tilde{Y}, U, W) &= \max_{1 \le l \le N} \max_{1 \le i \le n} \left\{ \lambda_i |\tilde{x}_{li} - \tilde{z}_{li}|, \theta_i |\tilde{y}_{li} - \tilde{z}_{li}| \right\} \\ &= \max_{1 \le l \le N} \max_{1 \le i \le n} \left\{ \lambda_i \left| \tilde{x}_{li} - \left(\sum_{(i,j) \in E} a_{ij} \tilde{x}_{lj} + u_{li} \right) \right|, \theta_i \left| \tilde{y}_{li} \right. \right. \\ &\quad \left. \left. - \left(\sum_{(i,j) \in E} a_{ij} \tilde{x}_{lj} + u_{li} \right) \right| \right\} \end{aligned}$$

Then the proposed optimization problem can be written as

$$\min_{W \in \mathbb{R}^{n \times n}(G)} J_\infty(\tilde{X}, \tilde{Y}, U, W)$$

and its optimal solution can be denoted by

$$W^* = \arg \min_{W \in \mathbb{R}^{n \times n}(G)} J_\infty(\tilde{X}, \tilde{Y}, U, W)$$

The formulation does not consider the case where some data are outliers in $\tilde{X}$, $\tilde{Y}$, and U. To prescreen the observation data, one can use some machine learning approach such as support vector machine for cluster data pattern recognition. This will serve as the first step for the SEM-based approach by filtering out some possible outliers in N measurement data before feeding them into the proposed LSM approach.

Lastly, the optimal solution W^* to these optimization problems may not be unique. In fact, there could exist infinitely many optimal solutions to J_1, J_2, or J_∞. Hence, to narrow down the search for the optimal solution, one can consider some additional constraint on W such as $\|W^*\|_{\mathrm{F}} = \min$ or $\|W^*\|_{\mathrm{F}}$ is a constant.

Chapter 4

A cooperative game approach to modeling the positive impact of variable correlation

Although the model in (2.6) gives a fundamental structure for variable correlation, there are some related open questions.

Q1. The previously proposed LSM approach has a key assumption that the information of $\tilde{x}_j$ is available for conducting the LSM evaluation as shown in the three proposed cost functions. According to (2.6), the information x_i is modeled as a linear combination of x_j, $j \in N_i$, where $N_i = \{j \in V : (i,j) \in E\}$. So is $\tilde{x}_i$ in terms of $\tilde{x}_j$. In real-world scenarios, the information of x_j may not be directly accessible to node i due to some impact from environmental disturbance, model uncertainty, data error, data privacy, or data security, and node i can only use some local sensors to *predict* x_j rather than directly measure x_j. In this case, how can the parameter matrix W be identified in the model (2.6) with a possibly imprecise prediction on x_j?

Q2. In addition, when some measurements are imperfect (e.g., y) and some data information is not directly accessible at the same time(e.g., x_j), how can the value of the entries for W be estimated?

To answer these questions, the previously proposed LSM approach is extended by proposing a coupled optimization problem that includes both the LSM evaluation and unknown variable prediction. Here the focus is on (2.12) or (2.13) to develop a prediction algorithm beginning with X. Then this prediction is embedded into the proposed LSM problem.

Consider (2.13). Define

$$\mathcal{X} = \{X^* \in \mathbb{R}^{N\times n} : X^* = \arg\min_X \|X - XW^{\mathrm{T}} - U\|_{\mathrm{F}}\}$$

Clearly $\|X-XW^{\mathrm{T}}-U\|_{\mathrm{F}} \geq 0$ and $\|X-XW^{\mathrm{T}}-U\|_{\mathrm{F}} = 0$ if and only if $X = XW^{\mathrm{T}}+U$. If x_j contains some uncertainty Δx_j, then the SEM model becomes $x = W(x+\Delta x)+u$. In the case of measurement matrices X, U with the uncertainty matrix ΔX,

$$X = (X + \Delta X)W^{\mathrm{T}} + U$$

and hence, $\|X - XW^{\mathrm{T}} - U\|_{\mathrm{F}} = \|(\Delta X)W^{\mathrm{T}}\|_{\mathrm{F}}$. Alternatively, if y contains some uncertainty Δy, then the SEM model becomes $x + \Delta y = Wx + u$. In the case of measurement matrices, X, U with the uncertainty matrix ΔY,

$$X + \Delta Y = XW^{\mathrm{T}} + U$$

In this case, $\|X - XW^{\mathrm{T}} - U\|_{\mathrm{F}} = \|\Delta Y\|_{\mathrm{F}}$. Finally, if the both x_j and y contain uncertainties, then $x + \Delta y = W(x + \Delta x) + u$. Similarly,

$$X + \Delta Y = (X + \Delta X)W^{\mathrm{T}} + U$$

Therefore, $\|X - XW^{\mathrm{T}} - U\|_{\mathrm{F}} = \|(\Delta X)W^{\mathrm{T}} - \Delta Y\|_{\mathrm{F}}$.

Lemma 3. *$X \in \mathcal{X}$ if and only if $(X - XW^{\mathrm{T}} - U)(I_n - W) = 0$.*

Proof: Note that by Lemma 2,

$$\|X - XW^{\mathrm{T}} - U\|_{\mathrm{F}} = \|\mathrm{vec}(X(I_n - W^{\mathrm{T}}) - U)\|_2 = \|((I_n - W) \otimes I_N)\mathrm{vec}\, X - \mathrm{vec}\, U\|_2$$

Now consider the LSM problem

$$\min_{\mathrm{vec}\, X} \|((I_n - W) \otimes I_N)\mathrm{vec}\, X - \mathrm{vec}\, U\|_2$$

Then $\mathrm{vec}\, X$ is a solution to this LSM problem if and only if

$$((I_n - W) \otimes I_N)^{\mathrm{T}}[((I_n - W) \otimes I_N)\mathrm{vec}\, X - \mathrm{vec}\, U] = 0$$

that is,

$$(((I_n - W)^{\mathrm{T}}(I_n - W)) \otimes I_N)\mathrm{vec}\, X - ((I_n - W)^{\mathrm{T}} \otimes I_N)\mathrm{vec}\, U = 0$$

where the fact that $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$ and $(A \otimes B)^{\mathrm{T}} = (A^{\mathrm{T}} \otimes B^{\mathrm{T}})$ is used.

Next, using the fact that $(B^{\mathrm{T}} \otimes A)\mathrm{vec}\, X = \mathrm{vec}(AXB)$, it can be seen

$$\begin{aligned} 0 &= (((I_n - W)^{\mathrm{T}}(I_n - W)) \otimes I_N)\mathrm{vec}\, X - ((I_n - W)^{\mathrm{T}} \otimes I_N)\mathrm{vec}\, U \\ &= \mathrm{vec}\,(X(I_n - W)^{\mathrm{T}}(I_n - W)) - \mathrm{vec}\,(U(I_n - W)) \\ &= \mathrm{vec}(X(I_n - W)^{\mathrm{T}}(I_n - W) - U(I_n - W)) \end{aligned}$$

which implies that

$$X(I_n - W)^{\mathrm{T}}(I_n - W) - U(I_n - W) = 0$$

■

Let $V = I_n - W$. Then $X \in \mathcal{X}$ if and only if $XV^{\mathrm{T}}V = UV$. Define $P = V^{\mathrm{T}}V$ and $Q = UV$. Then $X(I_n - W)^{\mathrm{T}}(I_n - W) - U(I_n - W) = 0$ is equivalent to

$$XP = Q$$

or

$$(P \otimes I_N)\mathrm{vec}\, X = \mathrm{vec}\, Q$$

Note that $P \in \mathbb{R}^{n \times n}$ is symmetric and positive semidefinite.

Lemma 4. *For any $x, y \in \mathbb{R}^n$, define the inner product $< x, y >= y^{\mathrm{T}}x$. Let $A \in \mathbb{R}^{n \times m}$ and $B \in \mathbb{R}^{m \times n}$. Then*

$$\mathrm{trace}(AB) = (\mathrm{vec}\, A^{\mathrm{T}})^{\mathrm{T}}\mathrm{vec}\, B =< \mathrm{vec}\, A^{\mathrm{T}}, \mathrm{vec}\, B >$$

Next , let

$$\psi(X, U, W) = \frac{1}{2} < \mathrm{vec}\, X, (P \otimes I_N)\mathrm{vec}\, X > - < \mathrm{vec}\, X, \mathrm{vec}\, Q >$$

Then it follows from Lemma 4 that

$$\psi(X,U,W) = \frac{1}{2} < \operatorname{vec} X, \operatorname{vec}(XP) > - < \operatorname{vec} X, \operatorname{vec} Q > \tag{4.1}$$

$$= \frac{1}{2}\operatorname{trace}(X^{\mathrm{T}}XP) - \operatorname{trace}(X^{\mathrm{T}}Q) \tag{4.2}$$

$$= \frac{1}{2}\operatorname{trace}(X^{\mathrm{T}}XV^{\mathrm{T}}V) - \operatorname{trace}(X^{\mathrm{T}}UV) \tag{4.3}$$

$$= \frac{1}{2}\operatorname{trace}(X^{\mathrm{T}}X(I_n - W)^{\mathrm{T}}(I_n - W)) - \operatorname{trace}(X^{\mathrm{T}}U(I_n - W)) \tag{4.4}$$

$$= \frac{1}{2}\operatorname{trace}((I_n - W)X^{\mathrm{T}}X(I_n - W)^{\mathrm{T}}) - \operatorname{trace}((I_n - W)X^{\mathrm{T}}U) \tag{4.5}$$

Lemma 5. *$XP = Q$ if and only if $X = \arg\min_{\boldsymbol{X}\in\mathbb{R}^{N\times n}} \psi(\boldsymbol{X}, U, W)$.*

Lemma 6. *$X \in \mathcal{X}$ if and only if $X = \arg\min_{\boldsymbol{X}\in\mathbb{R}^{n}} \psi(\boldsymbol{X}, U, W)$.*

Hence, the mathematical model of positive impact of variable correlation can be characterized by a coupled optimization problem for simultaneous prediction and identification:

$$X^* = \arg\min_{X\in\mathbb{R}^{N\times n}} \psi(X, U, W^*) \tag{4.6}$$

$$W^* = \arg\min_{W\in\mathbb{R}^{n\times n}(G)} J_a(X^*, \tilde{Y}, U, W) \tag{4.7}$$

where $a \in \{1, 2, \infty\}$.

Next, a cooperative game is proposed based multiagent coordination strategy to solve the coupled optimization problem (4.6) and (4.7). Motivated by previous work on bio-inspired consensus [23], a group of M agents representing the candidate estimates for X and W is considered. Each agent is denoted by $\boldsymbol{X}^{(l)}$ and $\boldsymbol{W}^{(l)}$, $l = 1, \ldots, M$. The values of $\boldsymbol{X}^{(l)}$ and $\boldsymbol{W}^{(l)}$ are updated numerically by a cooperative game based, semi-distributed (i.e., distributed-centralized), two-step algorithm. The first step is a local update formula according to two descent search directions. The

second step is a cooperative game based agreement algorithm for synchronizing local updates.

Definition 4.0.1. *The derivative* $\frac{\partial}{\partial X} f(X)$ *of a scalar-valued function* $f(X)$ *of a matrix argument* X
$= [X_{(i,j)}]_{i=1,\ldots,m,j=1,\ldots,n} \in \mathbb{R}^{m\times n}$ *is the* $n \times m$ *matrix whole* (i,j)*th element is* $\frac{\partial f(X)}{\partial X_{(j,i)}}$.

In this thesis, two types of such algorithms are considered.

Type I. The first type considers the case where $a = 2$. In this case, the cost function $J_2(\cdot,\cdot,\cdot,W)$ is differentiable with respect to W. We propose a derivative-involved semi-distributed algorithm inspired by [23]. The first part of the algorithm is to update W in two steps:

$$\begin{aligned}\boldsymbol{W}_{2k+1}^{(l)} = \boldsymbol{W}_{2k}^{(l)} &+ \alpha_k^{(l)}\left[\arg\min_{W=\boldsymbol{W}_{2k}^{(m)},m\in N_c^{(l)}\cup\{l\}} J_2(\boldsymbol{X}_{2k}^{(l)},\tilde{Y},U,W) - \boldsymbol{W}_{2k}^{(l)}\right]\circ C \\ &- \beta_k^{(l)}\left[\frac{\partial}{\partial W}J_2(\boldsymbol{X}_{2k}^{(l)},\tilde{Y},U,W)\Big|_{W=\boldsymbol{W}_{2k}^{(l)}}\right]\circ C \qquad (4.8)\end{aligned}$$

$$\boldsymbol{W}_{2k+2}^{(l)} = \arg\min_{W}\left[\sum_{m\in N_c^{(l)}\cup\{l\}}\|W - \boldsymbol{W}_{2k+1}^{(m)}\|_{\mathrm{F}}^2\right] \qquad (4.9)$$

where $k = 0,1,2,\ldots$, $\{\alpha_k^{(l)}\},\{\beta_k^{(l)}\} \subset \mathbb{R}$ are some parameter sequences. The coupling topology between these candidates is described by a graph $G_c = (V_c, E_c)$, $V_c = \{1,\ldots,M\}$, $E_c = \{(i,j) : i,j \in V_c, i \neq j\}$, and $N_c^{(l)} = \{m \in V_c : (l,m) \in E_c\}$ denoting the neighboring set of agent l. In this thesis, we always assume that G_c is undirected and strongly connected.

In this two-step algorithm, the first step consists of two terms. The term

$$\arg\min_{W=\boldsymbol{W}_{2k}^{(m)},m\in N_c^{(l)}\cup\{l\}} J_2(\boldsymbol{X}_{2k}^{(l)},\tilde{Y},U,W) - \boldsymbol{W}_{2k}^{(l)}$$

uses the difference between the local best solution among the neighboring nodes and the current value of W to update the search for W. It can be interpreted as the optimal search along the minimum cost level direction, which is roughly close to the tangent direction of the cost level curve. The term

$$-\frac{\partial}{\partial W} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W)\big|_{W=\boldsymbol{W}_{2k}^{(l)}}$$

uses the opposite gradient direction to update the search for W. It can be viewed as the optimal search along the normal direction of the cost level curve. Then these updates among all the neighboring nodes will be aggregated to reach a consensus for the second step update of W.

The update of X is given by a similar two-step algorithm:

$$\begin{aligned}\boldsymbol{X}_{2k+1}^{(l)} = \boldsymbol{X}_{2k}^{(l)} + \gamma_k^{(l)} \Bigg[\arg \min_{X=\boldsymbol{X}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} \psi(X, U, \boldsymbol{W}_{2k+2}^{(l)}) - \boldsymbol{X}_{2k}^{(l)} \Bigg] \\ - \delta_k^{(l)} \frac{\partial}{\partial X^{\mathrm{T}}} \psi(X, U, \boldsymbol{W}_{2k+2}^{(l)})\Big|_{X=\boldsymbol{X}_{2k}^{(l)}}\end{aligned} \tag{4.10}$$

$$\boldsymbol{X}_{2k+2}^{(l)} = \arg\min_X \Bigg[\sum_{m\in N_c^{(l)}\cup\{l\}} \|X - \boldsymbol{X}_{2k+1}^{(m)}\|_{\mathrm{F}}^2 \Bigg] \tag{4.11}$$

where $\{\gamma_k^{(l)}\}, \{\delta_k^{(l)}\} \subset \mathbb{R}$ are some parameter sequences.

The first issue of implementing Type I is to derive the explicit expression of (4.9) and (4.11). The following result solves this issue.

Lemma 7. *For any* $X_i \in \mathbb{R}^{n\times n}$, $i = 1, 2, \ldots, N$,

$$\arg\min_X \left[\sum_{i=1}^N \|X - X_i\|_{\mathrm{F}}^2\right] = \frac{1}{N}\sum_{i=1}^N X_i$$

Hence, (4.9) and (4.11) can be alternatively expressed as

$$\boldsymbol{W}_{2k+2}^{(l)} = \frac{1}{1+|N_c^{(l)}|}\sum_{m\in N_c^{(l)}\cup\{l\}} \boldsymbol{W}_{2k+1}^{(m)}$$
$$\boldsymbol{X}_{2k+2}^{(l)} = \frac{1}{1+|N_c^{(l)}|}\sum_{m\in N_c^{(l)}\cup\{l\}} \boldsymbol{X}_{2k+1}^{(m)}$$

The second issue of implementing Type I is to derive the explicit expression for $\frac{\partial}{\partial W}J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W)$ and $\frac{\partial}{\partial X^{\mathrm{T}}}\psi(X, U, \boldsymbol{W}_{2k+2}^{(l)})$. The following results give explicit expressions of these gradients.

Lemma 8. $\frac{\partial}{\partial X^{\mathrm{T}}}\psi(X, U, W) = XP - Q = X(I_n - W)^{\mathrm{T}}(I_n - W) - U(I_n - W)$.

Proof: Note that $\psi(X, U, W) = \frac{1}{2}\mathrm{trace}(X^{\mathrm{T}}XP) - \mathrm{trace}(X^{\mathrm{T}}Q)$. Next, note that the following identities hold:

$$\mathrm{trace}(AB) = \mathrm{trace}(BA) \tag{4.12}$$
$$\frac{\partial}{\partial X}\mathrm{trace}(AX) = A \tag{4.13}$$
$$\frac{\partial}{\partial X}\mathrm{trace}(AXBX^{\mathrm{T}}) = BX^{\mathrm{T}}A + B^{\mathrm{T}}X^{\mathrm{T}}A^{\mathrm{T}} \tag{4.14}$$

These lead to

$$\frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(AX^{\mathrm{T}}) = A$$
$$\frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(BXAX^{\mathrm{T}}) = BXA + B^{\mathrm{T}}XA^{\mathrm{T}}$$

Hence, it follows from the above formulas that

$$\frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(X^{\mathrm{T}}Q) = \frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(QX^{\mathrm{T}}) = Q$$
$$\frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(X^{\mathrm{T}}XP) = \frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(XPX^{\mathrm{T}}) = XP + XP = 2XP$$

Thus,

$$\frac{\partial}{\partial X^{\mathrm{T}}}\psi(X,U,W) = \frac{1}{2}\frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(X^{\mathrm{T}}XP) - \frac{\partial}{\partial X^{\mathrm{T}}}\mathrm{trace}(X^{\mathrm{T}}Q) = XP - Q$$

■

According to Lemma 8,

$$\frac{\partial}{\partial X^{\mathrm{T}}}\psi(X,U,\boldsymbol{W}_{2k+2}^{(l)})\bigg|_{X=\boldsymbol{X}_{2k}^{(l)}} = \boldsymbol{X}_{2k}^{(l)}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right)^{\mathrm{T}}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right) - U\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right)$$

and hence, (4.10) becomes

$$\begin{aligned}\boldsymbol{X}_{2k+1}^{(l)} = \boldsymbol{X}_{2k}^{(l)} + \gamma_k^{(l)}\Bigg[\arg\min_{X=\boldsymbol{X}_{2k}^{(m)},m\in N_c^{(l)}\cup\{l\}}\psi(X,U,\boldsymbol{W}_{2k+2}^{(l)}) - \boldsymbol{X}_{2k}^{(l)}\Bigg] \\ - \delta_k^{(l)}\boldsymbol{X}_{2k}^{(l)}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right)^{\mathrm{T}}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right) + \delta_k^{(l)}U\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right)\end{aligned}$$

Lemma 9. $\frac{\partial}{\partial W}J_2(X,Y,U,W) = 2X^{\mathrm{T}}XW^{\mathrm{T}}\Lambda - 2X^{\mathrm{T}}(X-U)\Lambda + 2X^{\mathrm{T}}XW^{\mathrm{T}}\Theta - 2X^{\mathrm{T}}(Y-U)\Theta$.

Proof: Note that

$$\begin{aligned} J_2(X,Y,U,W) &= \operatorname{trace}((X-U-XW^{\mathrm{T}})\Lambda(X-U-XW^{\mathrm{T}})^{\mathrm{T}}) \qquad (4.15)\\ &\quad + \operatorname{trace}((Y-U-XW^{\mathrm{T}})\Theta(Y-U-XW^{\mathrm{T}})^{\mathrm{T}})\\ &= \operatorname{trace}((X-U)\Lambda(X-U)^{\mathrm{T}}) + \operatorname{trace}(XW^{\mathrm{T}}\Lambda WX^{\mathrm{T}})\\ &\quad - \operatorname{trace}((X-U)\Lambda WX^{\mathrm{T}}) - \operatorname{trace}(XW^{\mathrm{T}}\Lambda(X-U)^{\mathrm{T}})\\ &\quad + \operatorname{trace}((Y-U)\Theta(Y-U)^{\mathrm{T}}) + \operatorname{trace}(XW^{\mathrm{T}}\Theta WX^{\mathrm{T}})\\ &\quad - \operatorname{trace}((Y-U)\Theta WX^{\mathrm{T}}) - \operatorname{trace}(XW^{\mathrm{T}}\Theta(Y-U)^{\mathrm{T}}) \qquad (4.16) \end{aligned}$$

Using (4.12)–(4.14) and the fact that $\operatorname{trace} A = \operatorname{trace} A^{\mathrm{T}}$, we have

$$\begin{aligned} \frac{\partial}{\partial W}\operatorname{trace}(XW^{\mathrm{T}}\Lambda WX^{\mathrm{T}}) &= \frac{\partial}{\partial W}\operatorname{trace}(\Lambda WX^{\mathrm{T}}XW^{\mathrm{T}})\\ &= X^{\mathrm{T}}XW^{\mathrm{T}}\Lambda + X^{\mathrm{T}}XW^{\mathrm{T}}\Lambda\\ &= 2X^{\mathrm{T}}XW^{\mathrm{T}}\Lambda \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial W}\operatorname{trace}(XW^{\mathrm{T}}\Theta WX^{\mathrm{T}}) &= 2X^{\mathrm{T}}XW^{\mathrm{T}}\Theta\\ \frac{\partial}{\partial W}\operatorname{trace}((X-U)\Lambda WX^{\mathrm{T}}) = \frac{\partial}{\partial W}\operatorname{trace}(X^{\mathrm{T}}(X-U)\Lambda W) &= X^{\mathrm{T}}(X-U)\Lambda\\ \frac{\partial}{\partial W}\operatorname{trace}(XW^{\mathrm{T}}\Lambda(X-U)^{\mathrm{T}}) = \frac{\partial}{\partial W}\operatorname{trace}((X-U)\Lambda WX^{\mathrm{T}}) &= X^{\mathrm{T}}(X-U)\Lambda\\ \frac{\partial}{\partial W}\operatorname{trace}((Y-U)\Theta WX^{\mathrm{T}}) &= X^{\mathrm{T}}(Y-U)\Theta\\ \frac{\partial}{\partial W}\operatorname{trace}(WX^{\mathrm{T}}\Theta(Y-U)) &= X^{\mathrm{T}}(Y-U)\Theta \end{aligned}$$

Hence,

$$\frac{\partial}{\partial W} J_2(X, Y, U, W) = 2X^{\mathrm{T}} X W^{\mathrm{T}} \Lambda - 2X^{\mathrm{T}}(X - U)\Lambda + 2X^{\mathrm{T}} X W^{\mathrm{T}} \Theta - 2X^{\mathrm{T}}(Y - U)\Theta$$

■

According to Lemma 9,

$$\begin{aligned}\frac{\partial}{\partial W} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W)\Big|_{W=\boldsymbol{W}_{2k}^{(l)}} = {} & 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} (\boldsymbol{W}_{2k}^{(l)})^{\mathrm{T}} \Lambda - 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} (\boldsymbol{X}_{2k}^{(l)} - U)\Lambda \\ & + 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} (\boldsymbol{W}_{2k}^{(l)})^{\mathrm{T}} \Theta - 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} (\tilde{Y} - U)\Theta\end{aligned}$$

and hence, (4.8) becomes

$$\begin{aligned}\boldsymbol{W}_{2k+1}^{(l)} = {} & \boldsymbol{W}_{2k}^{(l)} + \alpha_k^{(l)} \left[\arg \min_{W=\boldsymbol{W}_{2k}^{(m)}, m \in N_c^{(l)} \cup \{l\}} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W) - \boldsymbol{W}_{2k}^{(l)} \right] \circ C \\ & - 2\beta_k^{(l)} \Big[(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} (\boldsymbol{W}_{2k}^{(l)})^{\mathrm{T}} \Lambda - (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} (\boldsymbol{X}_{2k}^{(l)} - U)\Lambda \\ & + (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} (\boldsymbol{W}_{2k}^{(l)})^{\mathrm{T}} \Theta - (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} (\tilde{Y} - U)\Theta \Big] \circ C\end{aligned}$$

The third issue of implementing Type I is to determine $\{\alpha_k^{(l)}\}$, $\{\beta_k^{(l)}\}$, $\{\gamma_k^{(l)}\}$, and $\{\delta_k^{(l)}\}$. One way of determining these parameter sequences by using the steepest

descent idea. First, it follows from (4.15) that for any $K, L \in \mathbb{R}^{n\times n}$ and any $\alpha, \beta \in \mathbb{R}$,

$$\begin{aligned} J_2(X,Y,U,W+\alpha K+\beta L) = {} & \mathrm{trace}((X-U)\Lambda(X-U)^{\mathrm{T}}) \\ & + \mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}}) \\ & - \mathrm{trace}((X-U)\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}}) \\ & - \mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda(X-U)^{\mathrm{T}}) \\ & + \mathrm{trace}((Y-U)\Theta(Y-U)^{\mathrm{T}}) \\ & + \mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Theta(W+\alpha K+\beta L)X^{\mathrm{T}}) \\ & - \mathrm{trace}((Y-U)\Theta(W+\alpha K+\beta L)X^{\mathrm{T}}) \\ & - \mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Theta(Y-U)^{\mathrm{T}}) \end{aligned}$$

Lemma 10. *For any $K, L \in \mathbb{R}^{n\times n}$ and any $\alpha, \beta \in \mathbb{R}$,*

$$\begin{aligned} & \frac{\partial}{\partial\alpha} J_2(X,Y,U,W+\alpha K+\beta L) \\ & = 2\alpha\cdot\mathrm{trace}(XK^{\mathrm{T}}(\Lambda+\Theta)KX^{\mathrm{T}}) + 2\beta\cdot\mathrm{trace}(XK^{\mathrm{T}}(\Lambda+\Theta)LX^{\mathrm{T}}) \\ & \quad + 2\mathrm{trace}(XK^{\mathrm{T}}(\Lambda+\Theta)WX^{\mathrm{T}}) \\ & \quad - 2\mathrm{trace}((X-U)\Lambda KX^{\mathrm{T}}) - 2\mathrm{trace}((Y-U)\Theta KX^{\mathrm{T}}) \\ & \frac{\partial}{\partial\beta} J_2(X,Y,U,W+\alpha K+\beta L) \\ & = 2\alpha\cdot\mathrm{trace}(XL^{\mathrm{T}}(\Lambda+\Theta)KX^{\mathrm{T}}) + 2\beta\cdot\mathrm{trace}(XL^{\mathrm{T}}(\Lambda+\Theta)LX^{\mathrm{T}}) \\ & \quad + 2\mathrm{trace}(XL^{\mathrm{T}}(\Lambda+\Theta)WX^{\mathrm{T}}) \\ & \quad - 2\mathrm{trace}((X-U)\Lambda LX^{\mathrm{T}}) - 2\mathrm{trace}((Y-U)\Theta LX^{\mathrm{T}}) \end{aligned}$$

Proof: Note that

$$\begin{aligned}
&\frac{\partial}{\partial\alpha}\mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}})\\
&=\mathrm{trace}\Big(\frac{\partial}{\partial\alpha}\big(X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}}\big)\Big)\\
&=\mathrm{trace}(XK^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}}+X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda KX^{\mathrm{T}})\\
&=2\alpha\cdot\mathrm{trace}(XK^{\mathrm{T}}\Lambda KX^{\mathrm{T}})+2\beta\cdot\mathrm{trace}(XK^{\mathrm{T}}\Lambda LX^{\mathrm{T}})+2\mathrm{trace}(XK^{\mathrm{T}}\Lambda WX^{\mathrm{T}})\\
&\frac{\partial}{\partial\alpha}\mathrm{trace}((X-U)\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}})=\mathrm{trace}((X-U)\Lambda KX^{\mathrm{T}})
\end{aligned}$$

Similarly,

$$\begin{aligned}
&\frac{\partial}{\partial\alpha}\mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Theta(W+\alpha K+\beta L)X^{\mathrm{T}})\\
&=2\alpha\cdot\mathrm{trace}(XK^{\mathrm{T}}\Theta KX^{\mathrm{T}})+2\beta\cdot\mathrm{trace}(XK^{\mathrm{T}}\Theta LX^{\mathrm{T}})+2\mathrm{trace}(XK^{\mathrm{T}}\Theta WX^{\mathrm{T}})\\
&\frac{\partial}{\partial\alpha}\mathrm{trace}((Y-U)\Theta(W+\alpha K+\beta L)X^{\mathrm{T}})=\mathrm{trace}((Y-U)\Theta KX^{\mathrm{T}})
\end{aligned}$$

and

$$
\begin{aligned}
&\frac{\partial}{\partial\beta}\mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}})\\
&=\mathrm{trace}\Big(\frac{\partial}{\partial\beta}\big(X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}}\big)\Big)\\
&=\mathrm{trace}(XL^{\mathrm{T}}\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}}+X(W+\alpha K+\beta L)^{\mathrm{T}}\Lambda LX^{\mathrm{T}})\\
&=2\alpha\cdot\mathrm{trace}(XL^{\mathrm{T}}\Lambda KX^{\mathrm{T}})+2\beta\cdot\mathrm{trace}(XL^{\mathrm{T}}\Lambda LX^{\mathrm{T}})+2\mathrm{trace}(XL^{\mathrm{T}}\Lambda WX^{\mathrm{T}})\\
&\frac{\partial}{\partial\beta}\mathrm{trace}((X-U)\Lambda(W+\alpha K+\beta L)X^{\mathrm{T}})=\mathrm{trace}((X-U)\Lambda LX^{\mathrm{T}})\\
&\frac{\partial}{\partial\beta}\mathrm{trace}(X(W+\alpha K+\beta L)^{\mathrm{T}}\Theta(W+\alpha K+\beta L)X^{\mathrm{T}})\\
&=\mathrm{trace}\Big(\frac{\partial}{\partial\beta}\big(X(W+\alpha K+\beta L)^{\mathrm{T}}\Theta(W+\alpha K+\beta L)X^{\mathrm{T}}\big)\Big)\\
&=\mathrm{trace}(XL^{\mathrm{T}}\Theta(W+\alpha K+\beta L)X^{\mathrm{T}}+X(W+\alpha K+\beta L)^{\mathrm{T}}\Theta LX^{\mathrm{T}})\\
&=2\alpha\cdot\mathrm{trace}(XL^{\mathrm{T}}\Theta KX^{\mathrm{T}})+2\beta\cdot\mathrm{trace}(XL^{\mathrm{T}}\Theta LX^{\mathrm{T}})+2\mathrm{trace}(XL^{\mathrm{T}}\Theta WX^{\mathrm{T}})\\
&\frac{\partial}{\partial\beta}\mathrm{trace}((Y-U)\Theta(W+\alpha K+\beta L)X^{\mathrm{T}})=\mathrm{trace}((Y-U)\Theta LX^{\mathrm{T}})
\end{aligned}
$$

Therefore, putting everything together yields

$$
\begin{aligned}
&\frac{\partial}{\partial\alpha}J_2(X,Y,U,W+\alpha K+\beta L)\\
&=2\alpha\cdot\mathrm{trace}(XK^{\mathrm{T}}(\Lambda+\Theta)KX^{\mathrm{T}})+2\beta\cdot\mathrm{trace}(XK^{\mathrm{T}}(\Lambda+\Theta)LX^{\mathrm{T}})\\
&\quad+2\mathrm{trace}(XK^{\mathrm{T}}(\Lambda+\Theta)WX^{\mathrm{T}})\\
&\quad-2\mathrm{trace}((X-U)\Lambda KX^{\mathrm{T}})-2\mathrm{trace}((Y-U)\Theta KX^{\mathrm{T}})\\
&\frac{\partial}{\partial\beta}J_2(X,Y,U,W+\alpha K+\beta L)\\
&=2\alpha\cdot\mathrm{trace}(XL^{\mathrm{T}}(\Lambda+\Theta)KX^{\mathrm{T}})+2\beta\cdot\mathrm{trace}(XL^{\mathrm{T}}(\Lambda+\Theta)LX^{\mathrm{T}})\\
&\quad+2\mathrm{trace}(XL^{\mathrm{T}}(\Lambda+\Theta)WX^{\mathrm{T}})\\
&\quad-2\mathrm{trace}((X-U)\Lambda LX^{\mathrm{T}})-2\mathrm{trace}((Y-U)\Theta LX^{\mathrm{T}})
\end{aligned}
$$

■

Now Lemma 10 is used to find out $\{\alpha_k^{(l)}\}$ and $\{\beta_k^{(l)}\}$. To this end, define

$$
\begin{aligned}
K_{2k}^{(l)} &= \arg \min_{W=\boldsymbol{W}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W) - \boldsymbol{W}_{2k}^{(l)} \\
L_{2k}^{(l)} &= -\frac{\partial}{\partial W} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W)\Big|_{W=\boldsymbol{W}_{2k}^{(l)}} \\
&= -2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} (\boldsymbol{W}_{2k}^{(l)})^{\mathrm{T}} \Lambda + 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} (\boldsymbol{X}_{2k}^{(l)} - U)\Lambda \\
&\quad - 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} (\boldsymbol{W}_{2k}^{(l)})^{\mathrm{T}} \Theta + 2(\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}} (\tilde{Y} - U)\Theta \\
A_{11,2k}^{(l)} &= \mathrm{trace}(\boldsymbol{X}_{2k}^{(l)} (K_{2k}^{(l)})^{\mathrm{T}} (\Lambda + \Theta) K_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
A_{12,2k}^{(l)} &= \mathrm{trace}(\boldsymbol{X}_{2k}^{(l)} (K_{2k}^{(l)})^{\mathrm{T}} (\Lambda + \Theta) L_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
A_{21,2k}^{(l)} &= \mathrm{trace}(\boldsymbol{X}_{2k}^{(l)} (L_{2k}^{(l)})^{\mathrm{T}} (\Lambda + \Theta) K_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
A_{22,2k}^{(l)} &= \mathrm{trace}(\boldsymbol{X}_{2k}^{(l)} (L_{2k}^{(l)})^{\mathrm{T}} (\Lambda + \Theta) L_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
B_{1,2k}^{(l)} &= \mathrm{trace}(\boldsymbol{X}_{2k}^{(l)} (K_{2k}^{(l)})^{\mathrm{T}} (\Lambda + \Theta)\, \boldsymbol{W}_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) - \mathrm{trace}((\boldsymbol{X}_{2k}^{(l)} - U)\Lambda K_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
&\quad - \mathrm{trace}((\tilde{Y} - U)\Theta K_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
B_{2,2k}^{(l)} &= \mathrm{trace}(\boldsymbol{X}_{2k}^{(l)} (L_{2k}^{(l)})^{\mathrm{T}} (\Lambda + \Theta)\, \boldsymbol{W}_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) - \mathrm{trace}((\boldsymbol{X}_{2k}^{(l)} - U)\Lambda L_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}}) \\
&\quad - \mathrm{trace}((\tilde{Y} - U)\Theta L_{2k}^{(l)} (\boldsymbol{X}_{2k}^{(l)})^{\mathrm{T}})
\end{aligned}
$$

Here $\alpha_k^{(l)}$ and $\beta_k^{(l)}$ are chosen so that $\boldsymbol{W}_{2k+1}^{(l)}$ minimizes $J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W)$ when W is replaced by $\boldsymbol{W}_{2k+1}^{(l)}$. Then

$$
\begin{aligned}
\frac{\partial}{\partial \alpha_k^{(l)}} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U,\, \boldsymbol{W}_{2k}^{(l)} + \alpha_k^{(l)} K_{2k}^{(l)} + \beta_k^{(l)} L_{2k}^{(l)}) = 0 \\
\frac{\partial}{\partial \beta_k^{(l)}} J_2(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U,\, \boldsymbol{W}_{2k}^{(l)} + \alpha_k^{(l)} K_{2k}^{(l)} + \beta_k^{(l)} L_{2k}^{(l)}) = 0
\end{aligned}
$$

namely

$$A^{(l)}_{11,2k}\alpha^{(l)}_k + A^{(l)}_{12,2k}\beta^{(l)}_k + B^{(l)}_{1,2k} = 0$$
$$A^{(l)}_{21,2k}\alpha^{(l)}_k + A^{(l)}_{22,2k}\beta^{(l)}_k + B^{(l)}_{2,2k} = 0$$

Using the Cramer's rule, these two equations can be solved for $\alpha^{(l)}_k$ and $\beta^{(l)}_k$:

$$\alpha^{(l)}_k = \frac{\begin{vmatrix} -B^{(l)}_{1,2k} & A^{(l)}_{12,2k} \\ -B^{(l)}_{2,2k} & A^{(l)}_{22,2k} \end{vmatrix}}{\begin{vmatrix} A^{(l)}_{11,2k} & A^{(l)}_{12,2k} \\ A^{(l)}_{21,2k} & A^{(l)}_{22,2k} \end{vmatrix}} = \frac{B^{(l)}_{2,2k}A^{(l)}_{12,2k} - B^{(l)}_{1,2k}A^{(l)}_{22,2k}}{A^{(l)}_{11,2k}A^{(l)}_{22,2k} - A^{(l)}_{21,2k}A^{(l)}_{12,2k}}$$

$$\beta^{(l)}_k = \frac{\begin{vmatrix} A^{(l)}_{11,2k} & -B^{(l)}_{1,2k} \\ A^{(l)}_{21,2k} & -B^{(l)}_{2,2k} \end{vmatrix}}{\begin{vmatrix} A^{(l)}_{11,2k} & A^{(l)}_{12,2k} \\ A^{(l)}_{21,2k} & A^{(l)}_{22,2k} \end{vmatrix}} = \frac{B^{(l)}_{1,2k}A^{(l)}_{21,2k} - B^{(l)}_{2,2k}A^{(l)}_{11,2k}}{A^{(l)}_{11,2k}A^{(l)}_{22,2k} - A^{(l)}_{21,2k}A^{(l)}_{12,2k}}$$

where $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ is a 2 by 2 determinant whose result is $ad - bc$.

To determine $\gamma^{(l)}_k$ and $\delta^{(l)}_k$, we will take a similar approach to the above method. First, it follows from (4.1) that for any $R, S \in \mathbb{R}^{N\times n}$ and any $\alpha, \beta \in \mathbb{R}$,

$$\psi(X + \alpha R + \beta S, U, W) = \frac{1}{2}\mathrm{trace}((X + \alpha R + \beta S)^{\mathrm{T}}(X + \alpha R + \beta S)P) - \mathrm{trace}((X + \alpha R + \beta S)^{\mathrm{T}}Q)$$

Lemma 11. *For any $R, S \in \mathbb{R}^{N\times n}$ and any $\alpha, \beta \in \mathbb{R}$,*

$$\begin{aligned}\frac{\partial}{\partial\alpha}\psi(X+\alpha R+\beta S,U,W) &= \alpha\cdot\mathrm{trace}(R^{\mathrm{T}}RP)+\beta\cdot\mathrm{trace}(R^{\mathrm{T}}SP)\\ &\quad +\mathrm{trace}(R^{\mathrm{T}}XP)-\mathrm{trace}(R^{\mathrm{T}}Q)\\ \frac{\partial}{\partial\beta}\psi(X+\alpha R+\beta S,U,W) &= \alpha\cdot\mathrm{trace}(S^{\mathrm{T}}RP)+\beta\cdot\mathrm{trace}(S^{\mathrm{T}}SP)\\ &\quad +\mathrm{trace}(S^{\mathrm{T}}XP)-\mathrm{trace}(S^{\mathrm{T}}Q)\end{aligned}$$

Proof: Note that

$$\begin{aligned}\frac{\partial}{\partial\alpha}\psi(X+\alpha R+\beta S,U,W) &= \frac{1}{2}\mathrm{trace}(R^{\mathrm{T}}(X+\alpha R+\beta S)P)\\ &\quad +\frac{1}{2}\mathrm{trace}((X+\alpha R+\beta S)^{\mathrm{T}}RP)-\mathrm{trace}(R^{\mathrm{T}}Q)\\ &= \mathrm{trace}(R^{\mathrm{T}}(X+\alpha R+\beta S)P)-\mathrm{trace}(R^{\mathrm{T}}Q)\\ &= \alpha\cdot\mathrm{trace}(R^{\mathrm{T}}RP)+\beta\cdot\mathrm{trace}(R^{\mathrm{T}}SP)+\mathrm{trace}(R^{\mathrm{T}}XP)\\ &\quad -\mathrm{trace}(R^{\mathrm{T}}Q)\\ \frac{\partial}{\partial\alpha}\psi(X+\alpha R+\beta S,U,W) &= \frac{1}{2}\mathrm{trace}(S^{\mathrm{T}}(X+\alpha R+\beta S)P)\\ &\quad +\frac{1}{2}\mathrm{trace}((X+\alpha R+\beta S)^{\mathrm{T}}SP)-\mathrm{trace}(S^{\mathrm{T}}Q)\\ &= \mathrm{trace}(S^{\mathrm{T}}(X+\alpha R+\beta S)P)-\mathrm{trace}(S^{\mathrm{T}}Q)\\ &= \alpha\cdot\mathrm{trace}(S^{\mathrm{T}}RP)+\beta\cdot\mathrm{trace}(S^{\mathrm{T}}SP)+\mathrm{trace}(S^{\mathrm{T}}XP)\\ &\quad -\mathrm{trace}(S^{\mathrm{T}}Q)\end{aligned}$$

■

Now Lemma 10 is used to find out $\{\gamma_k^{(l)}\}$ and $\{\delta_k^{(l)}\}$. Define

$$
\begin{aligned}
R_{2k}^{(l)} &= \arg \min_{X=\boldsymbol{X}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} \psi(X, U, \boldsymbol{W}_{2k+2}^{(l)}) - \boldsymbol{X}_{2k}^{(l)} \\
S_{2k}^{(l)} &= -\frac{\partial}{\partial X^{\mathrm{T}}}\psi(X, U, \boldsymbol{W}_{2k+2}^{(l)})\Big|_{X=\boldsymbol{X}_{2k}^{(l)}} = -\boldsymbol{X}_{2k}^{(l)}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right)^{\mathrm{T}}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right) \\
&\quad + U\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right) \\
P_{2k}^{(l)} &= \left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right)^{\mathrm{T}}\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right) \\
Q_{2k}^{(l)} &= U\left(I_n - \boldsymbol{W}_{2k+2}^{(l)}\right) \\
E_{11,2k}^{(l)} &= \mathrm{trace}((R_{2k}^{(l)})^{\mathrm{T}} R_{2k}^{(l)} P_{2k}^{(l)}) \\
E_{12,2k}^{(l)} &= \mathrm{trace}(R_{2k}^{(l)} S_{2k}^{(l)} P_{2k}^{(l)}) \\
E_{21,2k}^{(l)} &= \mathrm{trace}((S_{2k}^{(l)})^{\mathrm{T}} R_{2k}^{(l)} P_{2k}^{(l)}) \\
E_{22,2k}^{(l)} &= \mathrm{trace}((S_{2k}^{(l)})^{\mathrm{T}} S_{2k}^{(l)} P_{2k}^{(l)}) \\
F_{1,2k}^{(l)} &= \mathrm{trace}((R_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} P_{2k}^{(l)}) - \mathrm{trace}((R_{2k}^{(l)})^{\mathrm{T}} Q_{2k}^{(l)}) \\
F_{2,2k}^{(l)} &= \mathrm{trace}((S_{2k}^{(l)})^{\mathrm{T}} \boldsymbol{X}_{2k}^{(l)} P_{2k}^{(l)}) - \mathrm{trace}((S_{2k}^{(l)})^{\mathrm{T}} Q_{2k}^{(l)})
\end{aligned}
$$

Here we choose $\gamma_k^{(l)}$ and $\delta_k^{(l)}$ so that $\boldsymbol{X}_{2k+1}^{(l)}$ minimizes $\psi(X, U, \boldsymbol{W}_{2k+2}^{(l)})$ when X is replaced by $\boldsymbol{X}_{2k+1}^{(l)}$. Then we have

$$
\begin{aligned}
\frac{\partial}{\partial \gamma_k^{(l)}}\psi(\boldsymbol{X}_{2k}^{(l)} + \gamma_k^{(l)} R_{2k}^{(l)} + \delta_k^{(l)} S_{2k}^{(l)}, U, \boldsymbol{W}_{2k+2}^{(l)}) = 0 \\
\frac{\partial}{\partial \delta_k^{(l)}}\psi(\boldsymbol{X}_{2k}^{(l)} + \gamma_k^{(l)} R_{2k}^{(l)} + \delta_k^{(l)} S_{2k}^{(l)}, U, \boldsymbol{W}_{2k+2}^{(l)}) = 0
\end{aligned}
$$

namely

$$E_{11,2k}^{(l)}\gamma_k^{(l)} + E_{12,2k}^{(l)}\delta_k^{(l)} + F_{1,2k}^{(l)} = 0$$
$$E_{21,2k}^{(l)}\gamma_k^{(l)} + E_{22,2k}^{(l)}\delta_k^{(l)} + F_{2,2k}^{(l)} = 0$$

Again, using the Cramer's rule, we can solve these two equations for $\gamma_k^{(l)}$ and $\delta_k^{(l)}$:

$$\gamma_k^{(l)} = \frac{\begin{vmatrix} -F_{1,2k}^{(l)} & E_{12,2k}^{(l)} \\ -F_{2,2k}^{(l)} & E_{22,2k}^{(l)} \end{vmatrix}}{\begin{vmatrix} E_{11,2k}^{(l)} & E_{12,2k}^{(l)} \\ E_{21,2k}^{(l)} & E_{22,2k}^{(l)} \end{vmatrix}} = \frac{F_{2,2k}^{(l)}E_{12,2k}^{(l)} - F_{1,2k}^{(l)}E_{22,2k}^{(l)}}{E_{11,2k}^{(l)}E_{22,2k}^{(l)} - E_{21,2k}^{(l)}E_{12,2k}^{(l)}} \tag{4.17}$$

$$\delta_k^{(l)} = \frac{\begin{vmatrix} E_{11,2k}^{(l)} & -F_{1,2k}^{(l)} \\ E_{21,2k}^{(l)} & -F_{2,2k}^{(l)} \end{vmatrix}}{\begin{vmatrix} E_{11,2k}^{(l)} & E_{12,2k}^{(l)} \\ E_{21,2k}^{(l)} & E_{22,2k}^{(l)} \end{vmatrix}} = \frac{F_{1,2k}^{(l)}E_{21,2k}^{(l)} - F_{2,2k}^{(l)}E_{11,2k}^{(l)}}{E_{11,2k}^{(l)}E_{22,2k}^{(l)} - E_{21,2k}^{(l)}E_{12,2k}^{(l)}} \tag{4.18}$$

Type II. The second type considers the case where $a = 1$ or ∞. In this case, the cost function $J_a(\cdot,\cdot,\cdot,W)$ is *not* differentiable with respect to W. We propose a non-smooth semi-distributed algorithm inspired by [14]. The first part of the algorithm is to update W in two steps:

$$\begin{aligned}
\boldsymbol{W}_{2k+1}^{(l)} = \boldsymbol{W}_{2k}^{(l)} &+ \alpha_k^{(l)}\left[\arg\min_{W=\boldsymbol{W}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} J_a(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W) - \boldsymbol{W}_{2k}^{(l)}\right]\circ C \\
&+ \beta_k^{(l)}\left[\arg\min_{W=\boldsymbol{W}_{2s}^{(m)}, m\in N_c^{(l)}\cup\{l\}, s=0,1,\ldots,k} J_a(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W) - \boldsymbol{W}_{2k}^{(l)}\right]\circ C \\
\boldsymbol{W}_{2k+2}^{(l)} &= \arg\min_W\left[\sum_{m\in N_c^{(l)}\cup\{l\}} \|W - \boldsymbol{W}_{2k+1}^{(m)}\|_{\mathrm{F}}^2\right]
\end{aligned}$$

Here the term

$$\arg \min_{W=\boldsymbol{W}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} J_a(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W) - \boldsymbol{W}_{2k}^{(l)}$$

represents the difference between the local best solution among node l's neighbors at the moment and the node l's current state in G_c. The term

$$\arg \min_{W=\boldsymbol{W}_{2s}^{(m)}, m\in N_c^{(l)}\cup\{l\}, s=0,1,\ldots,k} J_a(\boldsymbol{X}_{2k}^{(l)}, \tilde{Y}, U, W) - \boldsymbol{W}_{2k}^{(l)}$$

represents the difference between the global best solution among node l's neighbors up to now and the node l's current state in G_c.

Since $J_a(\cdot,\cdot,\cdot,W)$, $a \in \{1,\infty\}$, is not differentiable with respect to W, one cannot use Lemma 10 to determine $\alpha_k^{(l)}$ and $\beta_k^{(l)}$. Instead a heuristic approach is taken by selecting small positive values for $\alpha_k^{(l)}$ and $\beta_k^{(l)}$.

Then the update of X is given by a similar two-step algorithm:

$$\begin{aligned}
\boldsymbol{X}_{2k+1}^{(l)} &= \boldsymbol{X}_{2k}^{(l)} + \gamma_k^{(l)}\left[\arg \min_{X=\boldsymbol{X}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} \psi(X, U, \boldsymbol{W}_{2k+2}^{(l)}) - \boldsymbol{X}_{2k}^{(l)}\right] \\
&\quad + \delta_k^{(l)}\left[\arg \min_{X=\boldsymbol{X}_{2s}^{(m)}, m\in N_c^{(l)}\cup\{l\}, s=0,1,\ldots,k} \psi(X, U, \boldsymbol{W}_{2k+2}^{(l)}) - \boldsymbol{X}_{2k}^{(l)}\right] \\
\boldsymbol{X}_{2k+2}^{(l)} &= \arg\min_X \left[\sum_{m\in N_c^{(l)}\cup\{l\}} \|X - \boldsymbol{X}_{2k+1}^{(m)}\|_{\mathrm{F}}^2\right]
\end{aligned}$$

The parameters $\gamma_k^{(l)}$ and $\delta_k^{(l)}$ can determined by a similar approach which results in (4.17) and (4.18) with one minor change on $S_{2k}^{(l)}$ given by

$$S_{2k}^{(l)} = \arg \min_{X=\boldsymbol{X}_{2s}^{(m)}, m\in N_c^{(l)}\cup\{l\}, s=0,1,\ldots,k} \psi(X, U, \boldsymbol{W}_{2k+2}^{(l)}) - \boldsymbol{X}_{2k}^{(l)}$$

Chapter 5

A non-cooperative game approach to modeling the negative impact on variable correlation

In the cooperative game, there are two types of uncertainty existing in the model (2.6): Δy_i and Δx_j. More importantly, it is assumed there is no deterministic or stochastic property or structure for them. They are just part of the information embedded in the measurement $\tilde{X}$ and $\tilde{Y}$. Once the relevant data for (2.6) is obtained, uncertainty may exist in the measurement and the proposed cost function $J_a(\tilde{X}, \tilde{Y}, U, W)$, $a \in \{1, 2, \infty\}$ was to minimize its impact on the model (2.6).

To model the non-cooperative game for (2.6), a different setup from the cooperative game is considered. In this setup, when each player sends its information to another player, they intentionally distort their information by adding extra perturbation [15]. In particular, each player $i \in \{1, \ldots, n\}$ sets the output

$$\tilde{y}_i = z_i + w_i$$

where w_i is independent and identically distributed (i.i.d.) with zero mean and variance σ_i^2.

Unlike [15], here z_i is not independent of neighboring state uncertainty Δx_j, $(i, j) \in E$. In fact, since $z_i = \sum_{(i,j)\in E} a_{ij}\tilde{x}_j + u_i$ and $\tilde{x}_j = x_j + \Delta x_j$, it follows

that

$$z_i = \sum_{(i,j)\in E} a_{ij}x_j + \sum_{(i,j)\in E} a_{ij}\Delta x_j + u_i$$

which leads to

$$\tilde{y}_i = \sum_{(i,j)\in E} a_{ij}x_j + \sum_{(i,j)\in E} a_{ij}\Delta x_j + w_i + u_i$$

Here x_j and u_i are both deterministic. The above equation reveals an interesting fact that $\tilde{y}_i$ has two types of uncertainty: w_i and $\sum_{(i,j)\in E} a_{ij}\Delta x_j$. The first uncertainty w_i is set up by player i itself while the second one, $\sum_{(i,j)\in E} a_{ij}\Delta x_j$, is induced by the neighboring state uncertainty, Δx_j.

In general, a player intends to protect its real information by adding a perturbation to publicly available data during nuclear escalation. This translates into the language in the model that the uncertainty level for both w_i and Δx_i can be manipulated. Each player can manipulate the uncertainty level for both w_i and Δx_i so that $\tilde{y}_i$ is distorted jointly by player i and all the other neighboring players j, $(i,j) \in E$.

To deal with this more involved case, assume Δx_j is i.i.d. with zero mean and variance χ_j^2, w_i and Δx_s are independent, and $i, s = 1, \ldots, n$. Furthermore, assume

1) the noise level for w_i can be intentionally manipulated so that σ_i^2 varies in the interval $[\sigma_{\min}^2, \sigma_{\min}^2 + \sigma_{\max}^2]$, where $\sigma_{\min}^2$ denotes the inherent minimal noise level and $\sigma_{\max}^2$ denotes the maximally manipulated noise level;

2) the noise level for χ_j^2 can be intentionally manipulated so that it varies in the interval $[\chi_{\min}^2, \chi_{\min}^2 + \chi_{\max}^2]$, where $\chi_{\min}^2$ denotes the inherent minimal noise level and $\chi_{\max}^2$ denotes the maximally manipulated noise level. Assume that $\sigma_{\min}^2$,

$\chi^2_{\min}$, $\sigma^2_{\max}$, and $\chi^2_{\max}$ are known.

Let $\mathbb{E}$ denote the expectation operator. In summary, the new model with stochastic uncertainty for the non-cooperative game becomes

$$z_i = \sum_{(i,j)\in E} a_{ij}\tilde{x}_j + u_i, \quad i = 1, \ldots, n \tag{5.1}$$

$$\tilde{x}_j = x_j + \Delta x_j \tag{5.2}$$

$$\tilde{y}_i = z_i + w_i \tag{5.3}$$

where w_i and Δx_j satisfy

$$\mathbb{E}[w_i] = 0, \quad \mathbb{E}[w_i^2] = \sigma_i^2, \quad \mathbb{E}[w_i w_j] = 0, \quad i \neq j \tag{5.4}$$

$$\mathbb{E}[\Delta x_j] = 0, \quad \mathbb{E}[(\Delta x_j)^2] = \chi_j^2, \quad \mathbb{E}[\Delta x_j \Delta x_i] = 0, \quad i \neq j \tag{5.5}$$

$$\mathbb{E}[w_i \Delta x_s] = 0, \quad i, s = 1, \ldots, n \tag{5.6}$$

Next, let $\mu_i, \kappa_i > 0$ be chosen later and consider N repeated measurements for each $\tilde{x}_i$, $\tilde{y}_i$, and u_i, denoted by $\tilde{x}_{li}$, $\tilde{y}_{li}$, and u_{li}. Similar to the cooperative game case,

we consider a LSM function given by

$$\begin{aligned}
J_g(\tilde{X},\tilde{Y},U,W) &= \sum_{l=1}^{N}\sum_{i=1}^{n}\mu_i\left[\tilde{y}_{li}-\Big(\sum_{(i,j)\in E}a_{ij}\tilde{x}_{lj}+u_{li}\Big)\right]^2 \\
&\quad+\sum_{l=1}^{N}\sum_{i=1}^{n}\kappa_i\left[\tilde{x}_{li}-\Big(\sum_{(i,j)\in E}a_{ij}\tilde{x}_{lj}+u_{li}\Big)\right]^2 \\
&= \mathrm{trace}((\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})\Phi(\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}) \\
&\quad+\mathrm{trace}((\tilde{X}-U-\tilde{X}W^{\mathrm{T}})\Psi(\tilde{X}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}) \\
&= \mathrm{trace}((\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})\Phi) \\
&\quad+\mathrm{trace}((\tilde{X}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{X}-U-\tilde{X}W^{\mathrm{T}})\Psi)
\end{aligned}$$

where $\Phi = \mathrm{diag}(\mu_1,\ldots,\mu_n)$ and $\Psi = \mathrm{diag}(\kappa_1,\ldots,\kappa_n)$. Since $\tilde{X}$ and $\tilde{Y}$ now are stochastic, we are interested in averaging out the stochastic effect and calculating

$$\mathbb{E}[J_g(\tilde{X},\tilde{Y},U,W)]$$

Lemma 12.

$$\begin{aligned}
\mathbb{E}[J_g(\tilde{X},\tilde{Y},U,W)] &= \mathrm{trace}((Y-U-XW^{\mathrm{T}})\Phi(Y-U-XW^{\mathrm{T}})^{\mathrm{T}}) \\
&\quad+\mathrm{trace}((X-U-XW^{\mathrm{T}})\Psi(X-U-XW^{\mathrm{T}})^{\mathrm{T}})+F(W)
\end{aligned}$$

where

$$\begin{aligned}
F(W) &= N\sum_{i=1}^{n}(\sigma_i^2\mu_i+\chi_i^2\kappa_i)+N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}(\Phi+\Psi)W) \\
&\quad-2N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi)
\end{aligned}$$

Proof: Note that

$$\begin{aligned} J_g(\tilde{X},\tilde{Y},U,W) = &\operatorname{trace}((\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})\Phi) \\ &+ \operatorname{trace}((\tilde{X}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{X}-U-\tilde{X}W^{\mathrm{T}})\Psi) \end{aligned}$$

and

$$\begin{aligned} (\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{Y}-U-\tilde{X}W^{\mathrm{T}}) &= (\tilde{Y}-U)^{\mathrm{T}}(\tilde{Y}-U) + W\tilde{X}^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ &\quad - 2(\tilde{Y}-U)^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ &= \tilde{Y}^{\mathrm{T}}\tilde{Y} + U^{\mathrm{T}}U - 2\tilde{Y}^{\mathrm{T}}U + W\tilde{X}^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ &\quad - 2\tilde{Y}^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} + 2U^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ (\tilde{X}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{X}-U-\tilde{X}W^{\mathrm{T}}) &= (\tilde{X}-U)^{\mathrm{T}}(\tilde{X}-U) + W\tilde{X}^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ &\quad - 2(\tilde{X}-U)^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ &= \tilde{X}^{\mathrm{T}}\tilde{X} + U^{\mathrm{T}}U - 2\tilde{X}^{\mathrm{T}}U + W\tilde{X}^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \\ &\quad - 2\tilde{X}^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} + 2U^{\mathrm{T}}\tilde{X}W^{\mathrm{T}} \end{aligned}$$

Since $\tilde{X} = X + \Delta X$ and $\tilde{Y} = Y + \Delta Y$, it follows that $\mathbb{E}[\tilde{X}] = X$ and $\mathbb{E}[\tilde{Y}] = Y$. Furthermore,

$$\begin{aligned} \mathbb{E}[\tilde{X}^{\mathrm{T}}\tilde{X}] &= \mathbb{E}[X^{\mathrm{T}}X + 2X^{\mathrm{T}}\Delta X + (\Delta X)^{\mathrm{T}}\Delta X] = X^{\mathrm{T}}X + N\cdot\operatorname{diag}(\chi_1^2,\cdots,\chi_n^2) \\ \mathbb{E}[\tilde{Y}^{\mathrm{T}}\tilde{Y}] &= \mathbb{E}[Y^{\mathrm{T}}Y + 2Y^{\mathrm{T}}\Delta Y + (\Delta Y)^{\mathrm{T}}\Delta Y] = Y^{\mathrm{T}}Y + N\cdot\operatorname{diag}(\sigma_1^2,\cdots,\sigma_n^2) \\ \mathbb{E}[\tilde{Y}^{\mathrm{T}}\tilde{X}] &= \mathbb{E}[Y^{\mathrm{T}}X + (\Delta Y)^{\mathrm{T}}X + Y^{\mathrm{T}}\Delta X + (\Delta Y)^{\mathrm{T}}\Delta X] = Y^{\mathrm{T}}X \end{aligned}$$

Then we have

$$\begin{aligned}
\mathbb{E}[(\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{Y}-U-\tilde{X}W^{\mathrm{T}})\Phi] &= Y^{\mathrm{T}}Y\Phi + N\cdot\mathrm{diag}(\sigma_1^2,\cdots,\sigma_n^2)\Phi \\
&\quad + U^{\mathrm{T}}U\Phi - 2Y^{\mathrm{T}}U\Phi \\
&\quad + WX^{\mathrm{T}}XW^{\mathrm{T}}\Phi \\
&\quad + N\cdot W\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Phi \\
&\quad - 2Y^{\mathrm{T}}XW^{\mathrm{T}}\Phi + 2U^{\mathrm{T}}XW^{\mathrm{T}}\Phi \\
&= \mathrm{trace}((Y-U-XW^{\mathrm{T}}) \\
&\quad \Phi(Y-U-XW^{\mathrm{T}})^{\mathrm{T}}) \\
&\quad + N\sum_{i=1}^{n}\sigma_i^2\mu_i \\
&\quad + N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Phi W)
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathbb{E}[(\tilde{X}-U-\tilde{X}W^{\mathrm{T}})^{\mathrm{T}}(\tilde{X}-U-\tilde{X}W^{\mathrm{T}})\Psi] &= X^{\mathrm{T}}X\Psi + N\cdot\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)\Psi \\
&\quad + U^{\mathrm{T}}U\Psi - 2X^{\mathrm{T}}U\Psi \\
&\quad + WX^{\mathrm{T}}XW^{\mathrm{T}}\Psi \\
&\quad + N\cdot W\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi \\
&\quad - 2X^{\mathrm{T}}XW^{\mathrm{T}}\Phi \\
&\quad - 2N\cdot\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi \\
&\quad + 2U^{\mathrm{T}}XW^{\mathrm{T}}\Psi \\
&= \mathrm{trace}((X-U-XW^{\mathrm{T}}) \\
&\quad \Psi(X-U-XW^{\mathrm{T}})^{\mathrm{T}}) \\
&\quad + N\sum_{i=1}^{n}\chi_i^2\kappa_i \\
&\quad + N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi W) \\
&\quad - 2N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi)
\end{aligned}$$

Hence,

$$\begin{aligned}
\mathbb{E}[J_g(\tilde{X},\tilde{Y},U,W)] &= \mathrm{trace}((Y-U-XW^{\mathrm{T}})\Phi(Y-U-XW^{\mathrm{T}})^{\mathrm{T}}) \\
&\quad + \mathrm{trace}((X-U-XW^{\mathrm{T}})\Psi(X-U-XW^{\mathrm{T}})^{\mathrm{T}}) \\
&\quad + N\sum_{i=1}^{n}\sigma_i^2\mu_i + N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Phi W) \\
&\quad + N\sum_{i=1}^{n}\chi_i^2\kappa_i + N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi W) \\
&\quad - 2N\cdot\mathrm{trace}(\mathrm{diag}(\chi_1^2,\cdots,\chi_n^2)W^{\mathrm{T}}\Psi)
\end{aligned}$$

■

Thus, $\mathbb{E}[J_g(\tilde{X}, \tilde{Y}, U, W)]$ consists of two parts: the deterministic LSM function

$$\begin{aligned}&\text{trace}((Y - U - XW^{\mathrm{T}})\Phi(Y - U - XW^{\mathrm{T}})^{\mathrm{T}})+\\&\text{trace}((X - U - XW^{\mathrm{T}})\Psi(X - U - XW^{\mathrm{T}})^{\mathrm{T}})\end{aligned}$$

and the cost due to the uncertainty impact $F(W)$. It is clear that the deterministic LSM is independent of σ_i^2 and χ_i^2. It is more used to describe the cooperative part of variable correlation as formulated before. On the other hand, $F(W)$ is related to σ_i^2 and χ_i^2. Hence, for the proposed non-cooperative game that can intentionally modify the variance of data, $F(W)$ is one metric which measures the degree of manipulation. Motivated by this observation consider the following general optimization problem

$$\min_A F(A)$$

where A can choose any element in $\mathbb{R}^{n\times n}$, not restricted to the ones who have the same topology as G.

Lemma 13. *Let* $A^* = \arg\min_A F(A)$. *Then*

$$F(A^*) = N\sum_{i=1}^{n}(\sigma_i^2\mu_i + \chi_i^2\kappa_i) - N\sum_{i=1}^{n}\frac{\chi_i^2\kappa_i^2}{\mu_i + \kappa_i}$$

Proof: Let $\mathbb{X} = \text{diag}(\chi_1^2, \cdots, \chi_n^2)$. Then the solution to the original optimization problem $\min_A F(A)$ is the same as the solution to the following optimization problem

$$\min_A \text{trace}\left[\mathbb{X}A^{\mathrm{T}}(\Phi + \Psi)A - 2\mathbb{X}A^{\mathrm{T}}\Psi\right]$$

Note that

$$\frac{\partial}{\partial A}\text{trace}[\mathbb{X}A^{\mathrm{T}}(\Phi+\Psi)A] = \frac{\partial}{\partial A}\text{trace}[(\Phi+\Psi)A\mathbb{X}A^{\mathrm{T}}] = 2\mathbb{X}A^{\mathrm{T}}(\Phi+\Psi)$$
$$\frac{\partial}{\partial A}\text{trace}[\mathbb{X}A^{\mathrm{T}}\Psi] = \frac{\partial}{\partial A}\text{trace}[\Psi A\mathbb{X}] = \frac{\partial}{\partial A}\text{trace}[\mathbb{X}\Psi A] = \mathbb{X}\Psi$$

Hence,

$$\frac{\partial}{\partial A}\text{trace}\left[\mathbb{X}A^{\mathrm{T}}(\Phi+\Psi)A - 2\mathbb{X}A^{\mathrm{T}}\Psi\right] = 2\mathbb{X}A^{\mathrm{T}}(\Phi+\Psi) - 2\mathbb{X}\Psi$$

Setting this equation to be zero yields

$$2\mathbb{X}A^{\mathrm{T}}(\Phi+\Psi) - 2\mathbb{X}\Psi = 0$$

and hence,

$$A^* = (\Phi+\Psi)^{-1}\Psi$$

Substituting this A^* into $F(A)$ yields

$$\begin{aligned} F(A^*) &= N\sum_{i=1}^{n}(\sigma_i^2\mu_i + \chi_i^2\kappa_i) - N\cdot\text{trace}(\mathbb{X}(\Phi+\Psi)^{-1}\Psi^2) \\ &= N\sum_{i=1}^{n}(\sigma_i^2\mu_i + \chi_i^2\kappa_i) - N\sum_{i=1}^{n}\frac{\chi_i^2\kappa_i^2}{\mu_i+\kappa_i} \end{aligned}$$

■

Now choose

$$\mu_i = \frac{1}{\sigma_i^2}, \quad \kappa_i = \frac{1}{\chi_i^2}$$

to normalize variance manipulation in the cost $\mathbb{E}[J_g(\tilde{X}, \tilde{Y}, U, W)]$ so that the first term in $F(A)$ becomes a constat,

$$N\sum_{i=1}^{n}(\sigma_i^2\mu_i + \chi_i^2\kappa_i) = N\sum_{i=1}^{n}(1+1) = 2nN$$

The rest term of $F(A)$ becomes

$$-N\sum_{i=1}^{n}\frac{\chi_i^2\kappa_i^2}{\mu_i+\kappa_i} = -N\sum_{i=1}^{n}\frac{\frac{1}{\chi_i^2}}{\frac{1}{\sigma_i^2}+\frac{1}{\chi_i^2}}$$

Motivated by the above formulation, we can define a non-cooperative game whose players can intentionally change σ_i^2 and χ_j^2 so that the variance of w_i and Δx_j are manipulated due to data privacy and security concerns.

$$\mathcal{G} = \langle V, ([1/\sigma^2, 1]^n, [1/\chi^2, 1]^n), (\mathcal{J}_i)_{i\in V}\rangle$$

is denoted as the game with a set of players $V = \{1, \ldots, n\}$, where each player $i \in V$ chooses its action pair (p_i, q_i) in its action set $([1/(\sigma^2_{\min} + \sigma^2_{\max}), 1/\sigma^2_{\min}]^n, [1/(\chi^2_{\min} + \chi^2_{\max}), 1/\chi^2_{\min}]^n)$ to minimize the following cost

$$\mathcal{J}_i((p_i, q_i), (p_{-i}, q_{-i})) = c_i\|(p_i, q_i)\|_\ell^\ell + \frac{1}{2nN}F(A^*)$$

where $c_i > 0$, $\ell \geq 1$, and $\sigma_i^2 \in [\sigma^2_{\min}, \sigma^2_{\min} + \sigma^2_{\max}]$, $\chi_i \in [\chi^2_{\min}, \chi^2_{\min} + \chi^2_{\max}]$,

$$\begin{aligned} p_i =& 1/\sigma_i^2 \\ q_i =& 1/\chi_i^2 \\ F(A^*) =& 2nN - N\sum_{i=1}^{n}\frac{\frac{1}{\chi_i^2}}{\frac{1}{\sigma_i^2}+\frac{1}{\chi_i^2}} \end{aligned}$$

Standard notation, (p_{-i}, q_{-i}), is used to denote the collection of action pairs for all players except i.

In summary, the non-cooperative game based approach is to solve a zero-sum game whose local optimization problem for each agent i is given by

$$\min_{(p_i,q_i)} \mathcal{J}_i((p_i, q_i), (p_{-i}, q_{-i})) \tag{5.7}$$

that is, one needs to solve the following n local optimization problems simultaneously

$$\begin{aligned}
&\min_{(p_1,q_1)} \mathcal{J}_1((p_1, q_1), (p_{-1}, q_{-1})) \\
&\min_{(p_2,q_2)} \mathcal{J}_2((p_2, q_2), (p_{-2}, q_{-2})) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\min_{(p_n,q_n)} \mathcal{J}_n((p_n, q_n), (p_{-n}, q_{-n}))
\end{aligned}$$

Note that $\|(x, y)\|_\ell = (|x|^\ell + |y|^\ell)^{1/\ell}$ for $x, y \in \mathbb{R}$. Then $\|(p_i, q_i)\|_\ell^\ell = |p_i|^\ell + |q_i|^\ell$. In this project, we consider $\ell \in \{1, 2\}$.

An approach similar to the Type II algorithm is used to develop a numerical optimization solver for (p_i, q_i). In particular, the following two-step algorithm ($l =$

$1, \ldots, M$) is proposed:

$$
\begin{aligned}
(p_{i,2k+1}^{(l)}, q_{i,2k+1}^{(l)}) &= (p_{i,2k}^{(l)}, q_{i,2k}^{(l)}) \\
&\quad + \alpha_k^{(l)} \Big[\arg \min_{(p_i,q_i)=(p_{i,2k}^{(m)}, q_{i,2k}^{(m)}), m \in N_c^{(l)} \cup \{l\}} \mathcal{J}_i((p_i, q_i), (p_{-i,2k}^{(l)}, q_{-i,2k}^{(l)})) \\
&\quad - (p_{i,2k}^{(l)}, q_{i,2k}^{(l)}) \Big] \\
&\quad + \delta_k^{(l)} \Big[\arg \min_{(p_i,q_i)=(p_{i,2s}^{(m)}, q_{i,2s}^{(m)}), m \in N_c^{(l)} \cup \{l\}, s=0,1,\ldots,k} \mathcal{J}_i((p_i, q_i), (p_{-i,2k}^{(l)}, q_{-i,2k}^{(l)})) \\
&\quad - (p_{i,2k}^{(l)}, q_{i,2k}^{(l)}) \Big] \\
(p_{i,2k+2}^{(l)}, q_{i,2k+2}^{(l)}) &= \arg \min_{(p_i,q_i)} \Big[\sum_{m \in N_c^{(l)} \cup \{l\}} \|(p_i, q_i) - (p_{i,2k+1}^{(m)}, q_{i,2k+1}^{(m)})\|_{\mathrm{F}}^2 \Big], \\
& \qquad i = 1, \ldots, n, \quad k = 0, 1, 2, \ldots
\end{aligned}
$$

Once the best result for (p_i, q_i) are obtained, the result will be used to 1) generate a set of measurements $\tilde{X}$ and $\tilde{Y}$ based on the best $\sigma_i^2 = 1/p_i$ and $\chi_i^2 = 1/q_i$, and 2) replace the diagonal elements in Φ and Ψ in the LSM function $J_g(\tilde{X}, \tilde{Y}, U, W)$ by setting $\mu_i = p_i$ and $\kappa_i = q_i$.

Next, this updated LSM function with new Φ and Ψ will be used to solve the following optimization problem

$$\min_{W \in \mathbb{R}^{n \times n}(G)} \mathbb{E}[J_g(\tilde{X}, \tilde{Y}, U, W)]$$

to obtain the best W^*. This can be done by using a similar two-step numerical

algorithm:

$$\begin{aligned}
\boldsymbol{W}_{2k+1}^{(l)} = \boldsymbol{W}_{2k}^{(l)} + \alpha_k^{(l)} \Bigg[\arg \min_{W=\boldsymbol{W}_{2k}^{(m)}, m\in N_c^{(l)}\cup\{l\}} \mathbb{E}[J_g(\tilde{X},\tilde{Y},U,W)] - \boldsymbol{W}_{2k}^{(l)} \Bigg] \circ C \\
+ \beta_k^{(l)} \Bigg[\arg \min_{W=\boldsymbol{W}_{2s}^{(m)}, m\in N_c^{(l)}\cup\{l\}, s=0,1,\ldots,k} \mathbb{E}[J_g(\tilde{X},\tilde{Y},U,W)] - \boldsymbol{W}_{2k}^{(l)} \Bigg] \circ C \\
\boldsymbol{W}_{2k+2}^{(l)} = \arg\min_{W} \Bigg[\sum_{m\in N_c^{(l)}\cup\{l\}} \|W - \boldsymbol{W}_{2k+1}^{(m)}\|_{\mathrm{F}}^2 \Bigg]
\end{aligned}$$

When implementing this algorithm, one needs the information of X and Y due to the fact that

$$\mathbb{E}[J_g(\tilde{X},\tilde{Y},U,W)]$$

contains the deterministic cost

$$\begin{aligned}
&\mathrm{trace}((Y-U-XW^{\mathrm{T}})\Phi(Y-U-XW^{\mathrm{T}})^{\mathrm{T}}) \\
+&\mathrm{trace}((X-U-XW^{\mathrm{T}})\Psi(X-U-XW^{\mathrm{T}})^{\mathrm{T}})
\end{aligned}$$

Note that

$$\tilde{Y} = \begin{bmatrix} \tilde{y}[1] & \tilde{y}[2] & \dots & \tilde{y}[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \tilde{y}_{11} & \tilde{y}_{12} & \cdots & \tilde{y}_{1n} \\ \tilde{y}_{21} & \tilde{y}_{22} & \cdots & \tilde{y}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{y}_{N1} & \tilde{y}_{N2} & \cdots & \tilde{y}_{Nn} \end{bmatrix}$$

$$\tilde{X} = \begin{bmatrix} \tilde{x}[1] & \tilde{x}[2] & \dots & \tilde{x}[N] \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \tilde{x}_{11} & \tilde{x}_{12} & \cdots & \tilde{x}_{1n} \\ \tilde{x}_{21} & \tilde{x}_{22} & \cdots & \tilde{x}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{x}_{N1} & \tilde{x}_{N2} & \cdots & \tilde{x}_{Nn} \end{bmatrix}$$

Then Y and X are given by

$$Y = \mathbb{E}[\tilde{Y}] = \begin{bmatrix} \mathbb{E}[\tilde{y}_{11}] & \mathbb{E}[\tilde{y}_{12}] & \cdots & \mathbb{E}[\tilde{y}_{1n}] \\ \mathbb{E}[\tilde{y}_{21}] & \mathbb{E}[\tilde{y}_{22}] & \cdots & \mathbb{E}[\tilde{y}_{2n}] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[\tilde{y}_{N1}] & \mathbb{E}[\tilde{y}_{N2}] & \cdots & \mathbb{E}[\tilde{y}_{Nn}] \end{bmatrix} = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \\ y_1 & y_2 & \cdots & y_n \\ \vdots & \vdots & \ddots & \vdots \\ y_1 & y_2 & \cdots & y_n \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \otimes \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}$$

$$X = \mathbb{E}[\tilde{X}] = \begin{bmatrix} \mathbb{E}[\tilde{x}_{11}] & \mathbb{E}[\tilde{x}_{12}] & \cdots & \mathbb{E}[\tilde{x}_{1n}] \\ \mathbb{E}[\tilde{x}_{21}] & \mathbb{E}[\tilde{x}_{22}] & \cdots & \mathbb{E}[\tilde{x}_{2n}] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}[\tilde{x}_{N1}] & \mathbb{E}[\tilde{x}_{N2}] & \cdots & \mathbb{E}[\tilde{x}_{Nn}] \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1 & x_2 & \cdots & x_n \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \otimes \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$$

where x_i and y_i can be approximated by using the measurement data $\tilde{x}_{ji}$ and $\tilde{y}_{ji}$ as

follows

$$x_i \approx \frac{1}{N}\sum_{j=1}^{N} \tilde{x}_{ji}$$
$$y_i \approx \frac{1}{N}\sum_{j=1}^{N} \tilde{y}_{ji}$$

Finally, let W^*_{pro} denote the estimated coefficient matrix via the proposed cooperative game approach and W^*_{con} denote the estimated coefficient matrix via the proposed non-cooperative game approach. Then the overall estimate to consider both pro and con correlation effects between variables is given by the following form

$$W = (1-d)W^*_{\text{pro}} + dW^*_{\text{con}} \tag{5.8}$$

where $d \in [0, 1]$ is an index representing the probabilistic chance of leaning toward the pro or con effect for W. For example, for the balanced case of 50% of the pro effect and 50% of the con effect, one can choose $d = 0.5$. Alternatively, for the case of 30% of the pro effect and 70% of the con effect, one can choose $d = 0.7$. One key aspect of determining such d for nuclear competing modeling is that the value of d may be dynamically changing due to the escalation process. In this case, $d = d_t$ is not fixed and needs to be adjusted within several possible choices. For example, d could be varying among 0.2, 0.5, and 0.8, depending on the situation assessment and human decision maker's perception. To address this issue, a human-cognition-in-the-loop decision-making method to dynamically update d will be introduced later..

Chapter 6

Simulations of Cooperative and Non-cooperative Games

In this section, the preliminary simulation details and results for the proposed multi-agent hybrid game are outlined. The software, MATLAB R2021b, is used to apply and test the proposed game approaches by means of randomly generated data.

The cooperative game was simulated as follows. First, an input $U = [U_{ij}]_{i=1,\dots,N,j=1,\dots,n}$ to the model is considered. For this simulation, $U \in \mathbb{R}^{N\times n}$ is a randomly generated matrix whose entries are randomly chosen in the interval $[1, 10]$. Here, n is is the number of escalation factors considered and N is the number of different measurements for those escalation factors. Continuing on, X, Y, and Z are all matrices that are initialized prior to the simulation where X, Y, and $Z \in \mathbb{R}^{N\times n}$. $X = [X_{ij}]_{i=1,\dots,N,j=1,\dots,n}$ is initialized then Z and Y are calculated using Equations (2.12) and (2.15) respectively. The initial values of the parameters outlined in this section are shown in Table 6.1. Next, the topology of C is shown in Figure 6.1 and $A = [a_{ij}]_{i,j=1,\dots,n}$ is initialized, where C and $A \in \mathbb{R}^{n\times n}$. Now, W can be calculated as the Hadamard product of A and C. Next, the algorithm uses a two-step optimization formulation where each agent l is in the set of M number of agents. The weights for each optimization, $\Lambda \in \mathbb{R}^{n\times n}$ and $\Theta \in \mathbb{R}^{n\times n}$, are the given. Here, Λ and Θ are diagonal matrices whose values on the diagonal are equal to λ and θ as shown in Table 6.1. Next, each of the parameter sequences, α, β, δ, and γ, was set to a small value. Lastly, the neighboring

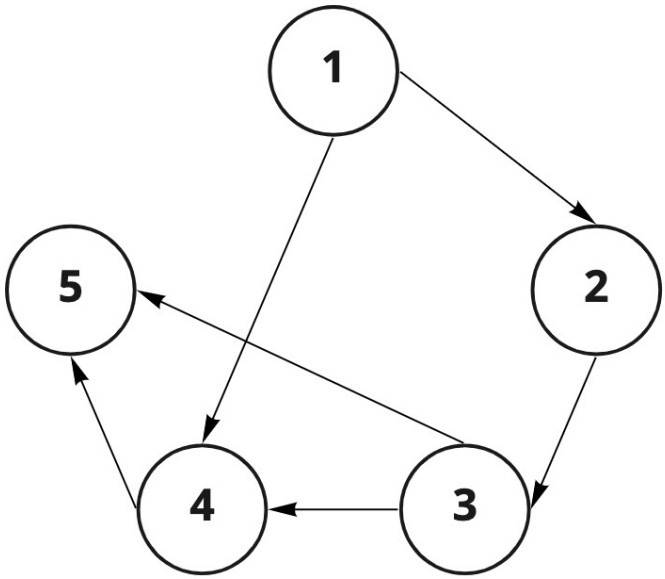

Figure 6.1: *Graph topology of C.*

set of agent l is N_c, and the number of elements in N_c is 2 for each agent.

The algorithm ran for k iterations where $k \leq$10000. In order to obtain a better estimation of the value for W_{pro}, the process was repeated until there were 20 runs that reached convergence. The topology of the matrix W_{pro} is shown in Figure 6.2. This figure shows that the proposed algorithm will converge and give credible information for random inputs. The topology shown here is equivalent to the SEM (2.1)–(2.5).

While random values were used for the initial simulation above, nuclear competing factors can be adapted to fit into the algorithm. Here, five competing factors ($n = 5$) are considered: defense budget, number of aircraft, ground vehicles, sea vessels, and military personnel, for the proposed model. It was assumed that during a nuclear competing escalation, the proposed SEM has the graph topology given by Figure 6.1, that is, the additional defense budget for countering nuclear escalation is mainly allo-

Table 6.1: *Parameter Inputs to the Proposed Algorithm*

Parameter Name	Parameter Value
U_{ij}	1:10
N	1
n	5
X_{ij}	1
a_{ij}	1
λ	1
θ	1
α	.001
β	.001
δ	.001
γ	.001
M	4
$\|N_c\|$	2

cated to aircraft for conducting a lot of surveillance and reconnaissance; the number of ground vehicles serving as possible reinforcement should correlate and match proportionally with the number of aircraft; the number of sea vessels is strongly related to both the defense budget and the number of ground vehicles; and the deployable military personnel are closely related to the number of ground vehicles and sea vessels, while the Air Force crew are hard to be recruited in a short period of time due to their existing surveillance and reconnaissance duties and the number limitation on pilots. Each competing factor is a component value in X for a given agent l. Using this information, the next simulation was run with the nuclear competing factors. The initial value of X is a random value drawn between the minimum and maximum parameter values shown in Table 6.2 and U is a zero matrix. In order to obtain a higher convergence rate for the algorithm, the values of X were normalized to values between 0 and 1. The scaling factors are also shown in Table 6.2. The topology of the algorithm was kept constant and the simulation ran until 20 convergence results were

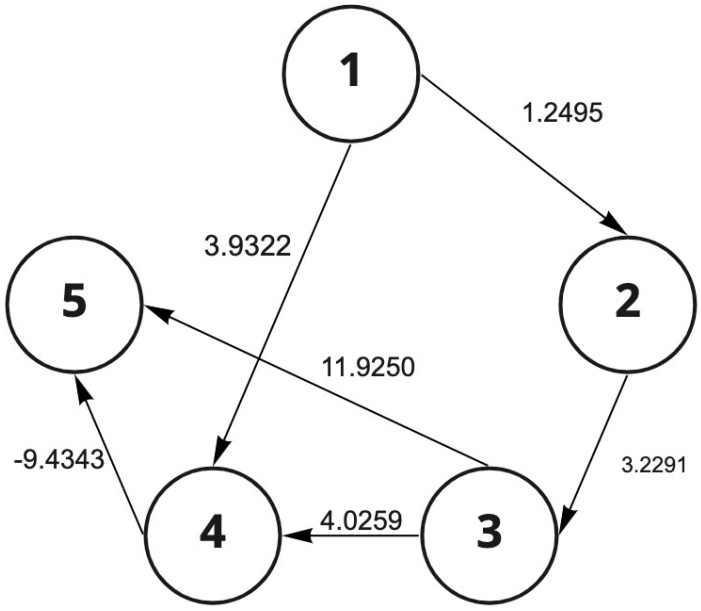

Figure 6.2: *Graph topology of* W_{pro} *with the weights on each edge.*

output. The average of these 20 runs was re-scaled according to the scaling factors in Table 6.2. Here, the resulting X in Equation (6.1) gives the convergent outcome of the players in the cooperative game. Specifically, X, shown in Equation (6.1), gives a steady-state value of nuclear competing factors for player 1 after gaming, indicating an equilibrium pattern for military deployment and resource allocation. The resulting matrix W_{pro} for this cooperative game is shown in Equation (6.2). Here, W_{pro} gives the weight matrix that optimizes the cooperative game for each player. Thus, the obtained model (2.6) has the following specific form to quantify the correlation

between these five factors:

$$X[2] = 2.4543 \cdot X[1]$$

$$X[3] = 1.2465 \cdot X[2]$$

$$X[4] = 5.8648 \cdot X[1] + 2.0378 \cdot X[3]$$

$$X[5] = 2.0442 \cdot X[3] + 0.8173 \cdot X[4]$$

where the coefficients represent the degree of correlation between relevant competing factors.

Table 6.2: *Nuclear Competing Factors X*

Parameter Label	Parameter Name	Parameter Value		Scaling Factor
		min	max	
Defense Budget	$X[1]$	6.34e8	7.405e11	1e12
Aircraft	$X[2]$	6.7e1	1.323e4	1e5
Ground Vehicles	$X[3]$	1.195e3	5.301e4	1e5
Sea Vessels	$X[4]$	0	4.90e2	1e3
Military Personnel	$X[5]$	2.593e5	2.2455e6	1e7

$$X = \begin{pmatrix} 3.46e10 \\ 1.292e4 \\ 1.305e4 \\ 3.882e2 \\ 5.809e6 \end{pmatrix} \tag{6.1}$$

$$W_{\text{pro}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 3.3849 & 0 & 0 & 0 & 0 \\ 0 & 0.94840 & 0 & 0 & \\ 5.6293 & 0 & 1.4917 & 0 & 0 \\ 0 & 0 & 1.8239 & 1.0384 & 0 \end{pmatrix} \tag{6.2}$$

Following the cooperative game, the non-cooperative game was simulated as follows. First each players action pair (p_i, q_i) was initialized as a random value in the interval $[0, 1]$, where $i = 1, \ldots, n$. Here, n is the number of players in the non-cooperative game. Then, the graph topology of C is setup. The topology is equal to the topology in the cooperative game and can be seen in Figure 6.1. Next the initial W was calculated, just as in the cooperative game. The initial values for these parameters and the remaining parameters that were used are shown in Table 6.3. Next, the algorithm uses a two-step optimization formulation to solve for the action pair. Here each agent l is in the set of M number of agents. Then, the algorithm uses another two-step optimization formulation to solve for W_{con}. To solve this optimization, x_i and y_i are formulated according to Equations (5.2) and (5.3).

The non-cooperative game ran for k iterations where $k \leq$10000. In order to obtain a better estimation of the value for W_{con}, the process was repeated until there were 20 runs that reached convergence. The resulting matrix W_{con} for this non-cooperative game is shown in Equation (6.3). Here, W gives the weight matrix that optimizes the non-cooperative game for each player.

Table 6.3: *Parameter Inputs to the Proposed Algorithm*

Parameter Name	Parameter Value
p_i	0:1
q_i	0:1
c_i	1
N	1
n	5
l	1
a_{ij}	1
α	.001
β	.001
δ	.001
M	4
$\|N_c\|$	2

$$W_{\text{con}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0.4736 & 0 & 0 & 0 & 0 \\ 0 & 0.53780 & 0 & 0 & \\ .04973 & 0 & 0.5431 & 0 & 0 \\ 0 & 0 & 0.5540 & 0.4938 & 0 \end{pmatrix} \tag{6.3}$$

Finally, with W_{pro} and W_{con}, Equation (5.8) can be used to find the value of W for agents competing in a mixed game consisting of both cooperative and non-cooperative gaming. Here, d is equal to .5, which weights the cooperative and non-cooperative game evenly. The value of W is shown in Equation (6.4).

$$W = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1.9292 & 0 & 0 & 0 & 0 \\ 0 & 0.74310 & 0 & 0 & \\ 3.0633 & 0 & 1.0174 & 0 & 0 \\ 0 & 0 & 1.1890 & 0.7661 & 0 \end{pmatrix} \tag{6.4}$$

The above results for W_{pro}, W_{con}, and W outline relationship among competing agents in a cooperative and non-cooperative gaming scenario. By comparing W_{pro} and W_{con}, it can be be seen that the values of W_{pro} are greater than W_{con}. This outcome was expected, as agents that partake in a cooperative game will form a greater positive correlation among one another. During this simulation, the value of W is the average of W_{pro}, W_{con}, since d is .5. To play a more dynamic game, the the value of d can be varied as the game is played. The simulations for Task 2 will explore this dynamic game by formulating d according to a multi-cue multi-choice model.

Chapter 7

Modeling the human behavior involved in decision making dynamics on different factors during escalation

The middle layer model utilizes a decision tree graph, a multirace network, and diffusion-based integrator dynamics together to mimic multiplayer decision making behavior with different degrees of adversarial structures among nuclear superpowers. The competing escalation dynamics are captured not only from an informatics perspective but also from a sociological perspective.

The human decision-making dynamics are modeled via multi-cue multi-choice tasks. The human decision-making behavior is incorporated into escalation dynamics by using a decision tree based multi-cue multi-choice task model to weigh in. To start with, we first introduce the multi-cue multi-choice task model for decision making. This multi-cue, multi-race model is a combination of a neural network with a multi-agent, multitask modeling idea. It describes the dynamic evolution of probabilistic information, quantified by a decision variable similar to Shannon Entropy in information theory, for making decision on different choices under difference cues. Before the model is introduced, some existing decision-making models are reviewed.

First, single-cue two-choice tasks are outlined. Here, mathematical representations of the human decision-making process analyze the simple two-alternative forced choice

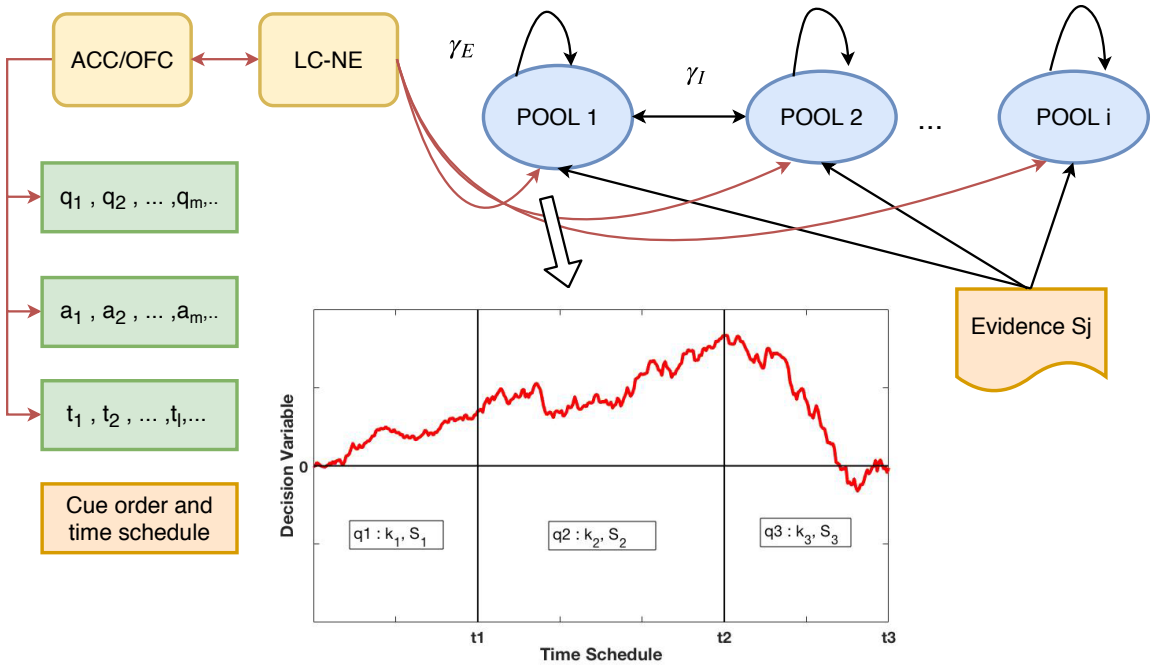

Figure 7.1: *Illustration of the MCMC task process. The model is governed by multiple O-U processes with external evidence S_j in multiple pools. The LE-NE method with ACC/OFC is used to adjust the cue order q_m, selection strategy parameters a_m, and time schedule t_l for multiple pools. γ_E and γ_I are excitatory and inhibitory gains that represent mutual inhibition between different pools.*

(2AFC) decision task [2]. Using the optimal sequential probability ratio test [6], the process on its continuum limit converges to a drift diffusion model if symmetric threshold is assumed. In this case, the decision variable for a noisy evidence can be modeled by a one-dimensional Wiener process bounded by positive and negative thresholds where an integrator accumulates the difference between two choices [2]. In order to reflect the effect of bounded accuracy and forgotten information, the drift diffusion integrators [2] are considered imperfect and leaky in the form of the Ornstein-Uhlenbeck (O-U) process. Although these models capture simple human decision making [2], they are only valid for 2AFC tasks, not for more general MCMC tasks [20].

Next, multi-cue two-choice tasks will be outlined. Here, real-world decision-making applications involve several cues [20]. One method of modeling multi-cue

tasks is to combine and integrate all cues in favor of each choice into a single source of evidence, ensuring that this source is used throughout the decision process. More involved treatment includes separate processes for each cue. Two important aspects of this approach are the order of considering the cues and the process time devoted to each cue. The time frame of this decision process can be divided into subintervals with different lengths during which the attention focus is only one cue. This method can address multiple cue issues. However, it is restricted to two alternative choices and fails with three or more choices.

Next, single-cue multi-choice tasks are outlined. One method of modeling multi-choice tasks is to separate leaky competing integrators and represent each choice with mutual inhibition [21, 7, 8, 17]. Each integrator gathers information in favor of or against the associated choice based on the value of the cue and the dynamics of each integrator governed by the O-U process. This method can overcome the drawback of two-choice modeling. However, it loses the ability to describe multi-cue tasks.

Lastly, multi-cue multi-choice tasks are outlined. In preliminary work [11, 9, 10], a new leaky integrator race model was proposed to capture the dynamics of strategy selection in multi-cue multi-choice (MCMC) tasks, continuously changing from comprehensive optimization (i.e., weighted-additive (WADD)), to simple heuristics (i.e., take-the-best (TTB)). This model, shown in Fig. 7.1, combines the multi-cue model, multi-choice task, reaction time, and order scheduling concept to evaluate the impact on the corresponding integrator at different time intervals. This scenario is quite common when a decision maker faces different situations and prioritizes some tasks by weighing the associated choices based on different cues. A neuromorphic approach, the locus coeruleus-norepinephrine (LC-NE) method [9], was used in this model to influence the decision-making outcome by controlling the excitatory and inhibitory gains signaled from the orbitofrontal cortex (OFC) and the anterior cingulate cor-

tex (ACC) in the brain. Although this theoretical model includes physiological and psychological impacts on human decision making in an innovative way; it remains unclear whether it traces real human decision making behavior under stress.

A MCMC task based method is proposed to determine d in (5.8) in an intelligent and adaptive way. That is, the determination of d is through deep reinforcement learning and the value of d is updated through an adaptive, iterative way to keep up with the changes of MCMC task conditions and competing escalation situations. Specifically, d is updated by a discrete MCMC model for mimicking human decision making. To derive such a model, a fundamental 2AFC task is reviewed.

A review of continuous-time 2AFC Models is as follows. Assume that for a task there are two choices S1 and S2 and evidence e presented in favor or against each choice. Conditional probabilities $p(e \mid S1)$ and $p(e \mid S2)$ are probabilities (with mean μ and variance σ^2) of observing evidence e under the occurrence of $S1$ or $S2$. Then the Bayes law gives

$$p(S1 \mid e) = \frac{p(e \mid S1)p(S1)}{p(e)}$$

Define the likelihood ratio as

$$LR(e) = \frac{p(e \mid S1)}{p(e \mid S2)}$$

Assuming independent evidences we have

$$LR(e) = \frac{p(S1 \mid e_1 \dots e_N)}{p(S2 \mid e_1 \dots e_N)} = \prod_{n=1}^{N} \frac{p(e_n \mid S1)}{p(e_n \mid S2)}$$

Taking logarithm on both sides yields

$$I^n = I^{(n-1)} + \log \frac{p(e_n \mid S1)}{p(e_n \mid S2)}$$

where

$$I^{(n-1)} = \sum_{n=1}^{n-1} \log \frac{p(e_n \mid S1)}{p(e_n \mid S2)}$$

Now let δI^r has finite mean μ and variance σ^2. Define a family of random functions, indexed by $M = 1, 2, \ldots$ of $t \in [0, T]$, where T is large enough, as follows

$$I^M(t) = \frac{1}{\sqrt{M}} \sum_{r=1}^{k} (\delta I^r - \mu) + \frac{1}{M} \sum_{r=1}^{k} \delta I^r \quad \text{where } k = \lfloor Mt/T \rfloor$$

where $\lfloor \cdot \rfloor$ denotes the floor function. For any M, $I^M(t)$ has the mean value of $\mu \lfloor t/T \rfloor$ and variance $\sigma^2 \lfloor t/T \rfloor$. According to Donsker invariance principle and the law of large number,

$$I^M \underset{f}{\rightarrow} \sigma \mathrm{W}(t) + \mu t \quad \text{as } M \rightarrow \infty$$

The above expression means converge in distribution and W is standard Wiener process. Hence, modeling the 2AFC task is given by the drift diffusion model (DDM)

$$dx = \mu dt + \sigma d\mathrm{W}(t)$$

where x is called *decision variable*, which can be viewed as a continuum version of I^M.

The general solution to the DDM with the initial condition $x(0) = 0$ is given by

$$p(x, t) = \mathcal{N}(\mu t, \sigma\sqrt{t}))$$

where $\mathcal{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and variance σ^2. If

$p_i(e_i|S_i) \sim \mathcal{N}(\mu_i, \sigma^2)$, then

$$p(e_i|S1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-(e_i - \mu_1)^2/2\sigma^2\}$$
$$p(e_i|S2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-(e_i - \mu_2)^2/2\sigma^2\}$$

Assume that the drift term is not constant and depends linearly to the decision variable in the DDM. Then the O-U process is modeled as

$$dx = (\mu + \lambda x)dt + \sigma d\mathrm{W}$$

For $\lambda < 0$ the process is stable and the time dependent probability of x converges to stationary normal distribution with mean of $-\mu/\lambda$ and variance of $-\sigma^2/2\lambda$.

The discrete-time version of 2AFC models can be derived based on the following setup.

- Markov process X_n, is a random variable over time interval $[0, t]$.
- The time is divided into subintervals of length τ.
- Process makes a step change at τ, 2τ, 3τ, ...
- The size of steps are assumed to be $\pm\Delta = \sqrt{t}$ with probabilities p_{ij}.
- The state space of the process is given by $S = \{-k\Delta, \ldots, -\Delta, 0, +\Delta, \ldots, +k\Delta\}$.
- $k = \lfloor t/\tau \rfloor$ is the number of time steps and $\pm k\Delta$ are the boundaries showing the decisions S1 and S2 respectively.

Using the above setup, the transition probability matrix of the discrete-time process

is given by the form

$$P = \left[\begin{array}{c|c} P_I & 0 \\ \hline R & Q \end{array}\right]$$

where the entries of P are described in detail by

	1	$m = 2k+1$	2	3	$\ldots$	$m-2$	$m-1$
absorbing S1 $\to 1$	1	0	0	0	$\ldots$	0	0
absorbing S2 $\to m$	0	1	0	0	$\ldots$	0	0
2	p_{21}	0	p_{21}	p_{23}	$\ldots$	0	0
3	0	0	p_{321}	p_{33}	$\ldots$	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$
$m-2$	0	0	0	0	$\ldots$	$p_{m-2,m-2}$	$p_{m-2,m-1}$
$m-1$	0	$p_{m-1,m}$	0	0	$\ldots$	$p_{m-1,m-2}$	$p_{m-1,m-1}$

Assuming the initial condition of Z, the probability and expected time to reach each decision are given by:

$$[p(S1), p(S2)] = Z(I-Q)^{-1}R$$

$$[\mathrm{E}(T/S1), \mathrm{E}(T/S2)] = \tau[Z(I-Q)^{-2}R] \circ [p(S1), p(S2)]$$

The increment of the process from time n to $n+1$ is given by:

$$X_{n+1} - X_n = Z_{n+1}, \quad n = 1, 2, 3, \ldots$$

$$X_{t/\tau} := \sum_{i=1}^{t/\tau} Z_i$$

where Z_i are independently and identically distributed. Assume $p[Z_i = +\Delta] = p[Z_i = -\Delta] = 0.5$. Then

$$\mathrm{E}(X_{t/\tau}) = 0, \quad \mathrm{Var}(X_{t/\tau}) = (t/\tau)\mathrm{Var}(Z_i) = t\Delta^2/\tau = t$$

Now letting $\tau \to 0$, this random walk model converges in distribution to standard Wiener process $\mathrm{W}(t)$ which from central limit theorem has normal distribution with zero mean and t variance.

Markov processes with continuous-time sets and continuous state space are called *diffusion processes* and the standard Wiener process is the simplest diffusion process. The Wiener process with drift is given by the form

$$V(t) = \mu t + \sigma \mathrm{W}(t)$$

where σ is a positive number. Also

$$dV(t) = V(t+\tau) - V(t) = \mu\tau + \sigma d\mathrm{W}(t)$$

The DDM can be modeled by Markov chains with transition probabilities

$$p_{i,i-1} = \frac{1}{2}\left(1 - \frac{\mu}{\sigma}\sqrt{\tau}\right)$$
$$p_{i,i+1} = \frac{1}{2}\left(1 + \frac{\mu}{\sigma}\sqrt{\tau}\right)$$

with $\sqrt{\tau} = \Delta/\sigma$ and $(-1/\sqrt{\tau}) \leq \mu/\sigma \leq (+1/\sqrt{\tau})$ to keep the p between zero and one. Note that here it is assumed $p_{ii} = 0$ or the process is not allowed to stay in the current stage at the next time step. If the process is allowed to stay at the same state at any time step, the Birth-Death process is characterized in continuous domain by the Ornstein-Uhlenbeck (O-U) process.

The velocity of information accumulation is assumed to be damped proportional to the current state while the new random information is added, given by the form

$$X(t+\tau) = (1 - \tau\gamma)X(t) + V(t+\tau)$$

or in the difference form

$$dX(t) = -\tau\gamma X(t) + V(t+\tau)$$

Let $V(t)$ be a Wiener process with drift δ and diffusion coefficient σ^2. Then

$$dX(t) = (\delta - \gamma X(t))\tau + \sigma d\mathrm{W}(t)$$

where the transition probabilities are given by

$$p_{i,j} = \begin{cases} \frac{1}{2\alpha}\left(1 - \frac{\delta-\gamma(-k\Delta+(i-1)\Delta)}{\sigma^2}\sqrt{\tau}\right) & \text{if } j-i=-1 \\ \frac{1}{2\alpha}\left(1 + \frac{\delta-\gamma(-k\Delta+(i-1)\Delta)}{\sigma^2}\sqrt{\tau}\right) & \text{if } j-i=+1 \\ 1-\frac{1}{\alpha} & \text{if } j=i \\ 0 & \text{otherwise} \end{cases}$$

$\Delta = \alpha\sigma\sqrt{\tau}$, and $\alpha > 1$ is a free parameter to restrict $p_{i,j}$ in between zero and one.

The discrete-time MCMC race model for descirbing MCMC tasks in nuclear competing dynamics can be derived in the following way. For given S choices, the process of evidence accumulation for MCMC tasks is governed by the O-U process:

$$d\mathbf{X}(t) = \mathbf{X}(t+\tau) - \mathbf{X}(t) = -\tau\Gamma(t)\mathbf{X}(t) + \mathbf{V}(t+\tau)$$

where $\mathbf{X}$ and $\mathbf{V}$ denote the vector form of X and V, respectively. Each drifted Wiener process V_i has the mean of δ_i and variance of σ_i^2

$$d\mathbf{X}(t) = (\delta\mathbf{e}_s - \Gamma\mathbf{X}(t))\tau + \Sigma d\mathbf{W}(t)$$

where $\mathbf{X} \in \mathbf{R}^S$ is column vector, Γ and $\Sigma \in \mathbf{R}^{S \times S}$ are S by S matrices and $\mathbf{e}_s$ is a $1 \times s$ column vector with all elements equal to one.

When discrete MCMC tasks are modeled, a few points that need to be considered are listed as follows.

- If the approach of discretization is the same as the two-choice model, conventional Sth order Markov chain of m states which has $O(m^S)$ state variables and therefore parameters is obtained.
- The number of transition probabilities (to be estimated) increases exponentially with respect to the order S (the number of choices) of the model.
- Circumventing this curse of dimensionality is done by autoregressive model estimation.

By definition an autoregressive process is a random process that its value in each time step is linearly dependent on its own previous values plus a stochastic term. The notion AR(p) indicates an autoregressive process of order p defined as:

$$X(t) = c + \sum_{i=1}^{p} \phi_i X_{t-i} + \mathrm{W}(t)$$

in which ϕ_i are the parameters of the model, c is a constant and $\mathrm{W}(t)$ is white noise. It is easy to see that the O-U process in discrete time can be modeled by AR(1) by rearranging the equation

$$X(t+1) = \mu t + (1-\gamma)X(t) + \sigma \mathrm{W}(t)$$

Let $\mathrm{x}^{(k)}(n) \in \mathbb{R}^m$ be the state probability distribution vector of the kth sequence at time n, where m denotes the number of cues. If the kth sequence is in state j with

probability one, then we say

$$\mathrm{x}^{(k)}(n) = \mathbf{e}_j = (0, \ldots, 0, \underbrace{1}_{j\text{th}}, 0, \ldots, 0)^{\mathrm{T}}$$

Moreover, we model the state transition as a compartmental system given by

$$\mathrm{x}^{(j)}(n+1) = \lambda_{jj} P^{(jj)} \mathrm{x}^{(j)}(n) + \sum_{k=1, k\neq j}^{S} P^{(jk)} \lambda_{jk} \mathrm{x}^{(k)}(n)$$

where $P^{(ij)} \in \mathbb{R}^{m\times m}$ is one step transition matrix of the state from the jth sequence to the state of the ith sequence,

$$\lambda_{jk} \geq 0, \quad 1 \leq j, k \leq S, \qquad \sum_{k=1}^{S} \lambda_{jk} = 1 \quad \text{for } j = 1, 2, \ldots, S$$

This model means that the state probability distribution of the jth chain at time $(n+1)$ depends only on the weighted average of itself, $P^{(jj)}\mathrm{x}^{(j)}(n)$, and other chains, $P^{(jk)}\mathrm{x}^{(k)}(n)$, at time n. Its vector form is given by

$$\mathbf{X}(n+1) \equiv \begin{bmatrix} \mathrm{x}^{(1)}(n+1) \\ \mathrm{x}^{(2)}(n+1) \\ \vdots \\ \mathrm{x}^{(S)}(n+1) \end{bmatrix} = \begin{bmatrix} \lambda_{11}P^{(11)} & \lambda_{12}P^{(12)} & \ldots & \lambda_{1S}P^{(1S)} \\ \lambda_{21}P^{(21)} & \lambda_{22}P^{(22)} & \ldots & \lambda_{2S}P^{(21)} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{S1}P^{(S1)} & \lambda_{S2}P^{(S1)} & \ldots & \lambda_{SS}P^{(SS)} \end{bmatrix} \begin{bmatrix} \mathrm{x}^{(1)}(n) \\ \mathrm{x}^{(2)}(n) \\ \vdots \\ \mathrm{x}^{(S)}(n) \end{bmatrix} \equiv Q\mathbf{X}(n) \tag{7.1}$$

Since all λ_{ij} are assumed to be nonnegative, the model only considers positive correlation among sequences. This means an increase in a state probability in any of the sequences at time n can only increase the state probabilities at time $n+1$.

Consider the following expression as a term being negatively correlated with x

$$\frac{1}{m-1}(\mathbf{1} - \mathrm{x}(n+1))$$

where $\mathbf{1} \in \mathbb{R}^m$ is a column vector of all ones, i.e., $\mathbf{1} = [1, \ldots, 1]^{\mathrm{T}} \in \mathbb{R}^m$, and $(m-1)$ is for normalization factor for number of state $m \geq 2$. Now the model (7.1) is extended to consider both negative and positive correlations as follows

$$\begin{bmatrix} \mathrm{x}^{(1)}(n+1) \\ \mathrm{x}^{(2)}(n+1) \\ \vdots \\ \mathrm{x}^{(S)}(n+1) \end{bmatrix} = \Lambda^{+} \begin{bmatrix} \mathrm{x}^{(1)}(n) \\ \mathrm{x}^{(2)}(n) \\ \vdots \\ \mathrm{x}^{(S)}(n) \end{bmatrix} + \frac{1}{m-1} \Lambda^{-} \begin{bmatrix} 1 - \mathrm{x}^{(1)}(n) \\ 1 - \mathrm{x}^{(2)}(n) \\ \vdots \\ 1 - \mathrm{x}^{(S)}(n) \end{bmatrix}$$

where

$$\Lambda^{+} = \begin{bmatrix} \lambda_{1,1} P^{(11)} & \lambda_{1,2} I & \ldots & \lambda_{1,S} I \\ \lambda_{2,1} I & \lambda_{2,2} P^{(22)} & \ldots & \lambda_{2,S} I \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{S,1} I & \lambda_{S,2} I & \ldots & \lambda_{SS} P^{(SS)} \end{bmatrix}$$

and

$$\Lambda^{-} = \begin{bmatrix} \lambda_{1,-1} P^{(11)} & \lambda_{1,-2} I & \ldots & \lambda_{1,-S} I \\ \lambda_{2,-1} I & \lambda_{2,-2} P^{(22)} & \ldots & \lambda_{2,-S} I \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{S,-1} I & \lambda_{S,-2} I & \ldots & \lambda_{S,-S} P^{(SS)} \end{bmatrix}$$

for $\lambda_{i,j} \geq 0$ for $i = 1, 2, \ldots, S$, $j = \pm 1, \pm 2, \ldots, \pm S$, and $\sum_{j=1}^{S} \lambda_{i,j} + \sum_{j=1}^{S} \lambda_{i,-j} = 1$. The matrix form of this extended model can be written as

$$\mathbf{X}(n+1) = \mathbf{H} \times \mathbf{X}(n) + \frac{1}{m-1} \mathbf{J} \times \mathbf{1} \equiv M_s \mathbf{X}(n) + \mathbf{b} \tag{7.2}$$

where $\mathbf{H} = \text{block}[H_{ij}]$ is a block matrix whose (i,j)th block is given by

$$H_{ij} = \begin{cases} (\lambda_{i,j} - \frac{\lambda i,-j}{m-1})P^{(ii)} & \text{if} \quad i = j \\ (\lambda_{i,j} - \frac{\lambda i,-j}{m-1})I & \text{otherwise} \end{cases}$$

and $\mathbf{J} = \text{block}[J_{ij}]$ is also a block matrix whose (i,j)th block is given by

$$J_{ij} = \begin{cases} \lambda_{i,-j}P^{(ii)} & \text{if} \quad i = j \\ \lambda_{i,-j}I & \text{otherwise} \end{cases}$$

Recursively using this model,

$$\mathbf{X}(n+1) = M_S^{(n+1)}\mathbf{X}(0) + \sum_{k=0}^{n} M_S^k \mathbf{b}$$

where $M_S^0 = I$.

If for certain matrix norm $\|\cdot\|$, we have $||M_S|| < 1$, then this extended model reaches a stationary distribution. For instance by considering $||\cdot||_\infty$ norm which is defined as

$$||M||_\infty = \max_i \left(\sum_{j=1}^{n} |M_{ij}| \right)$$

for some matrix $M = [M_{ij}]$, we have

$$\lim_{n\to\infty} \mathbf{X}(n) = \lim_{n\to\infty} \sum_{k=0}^{n} M_S^k \mathbf{b} = (I - M_S)^{-1}\mathbf{b}$$

Also note that

$$||M_S||_\infty \leq \max_{1\leq k\leq S} \left\{ m \left| \lambda_{k,k} - \frac{\lambda_{k,-k}}{m-1} \right| + \sum_{k\neq i} \left| \lambda_{k,i} - \frac{\lambda_{k,-i}}{m-1} \right| \right\}$$

which helps control the rate of convergence by setting the right hand side less than some

specific value $0 < c < 1$.

To estimate the parameters of this model, the final stationary distribution, namely $\hat{\mathbf{x}} = (\hat{\mathbf{x}}^{(1)}, \ldots, \hat{\mathbf{x}}^{(S)})$, and state transition probability matrices $P^{(ii)}$, should be given. These parameters can be calculated from previously recorded data of the sequences by solving the following constrained optimization problem

$$\min_{\Lambda^+, \Lambda^-} \sum_{j=1}^{S} \sum_{k=1}^{S} \|b_{j,k} - \hat{\mathbf{x}}^{(j)}\|_1 \tag{7.3}$$

where $\|\cdot\|_1$ is the 1-norm,

$$b_{j,k} = \sum_{k=1}^{S} \left((\lambda_{j,k} - \frac{\lambda_{j,-k}}{m-1}) \Delta_{j,k} \hat{\mathbf{x}}^{(k)} + \frac{1}{m-1} \lambda_{j,k} \Delta_{j,k} \mathbf{1} \right)$$

and

$$\Delta_{jk} = \begin{cases} P^{(jj)} & \text{if } j = k \\ I & \text{if } j \neq k \end{cases}$$

subject to the following constraints

$$1 - \sum_{k=1}^{S} \lambda_{j,k} = 0, \quad \forall j = 1, 2, \ldots, S \tag{7.4}$$

$$1 - \sum_{k=1}^{S} \lambda_{j,-k} = 0, \quad \forall j = 1, 2, \ldots, S \tag{7.5}$$

$$-\lambda_{j,k} \leq 0, \quad \forall k = \pm 1, \ldots, \pm S, \quad j = 1, 2, \ldots S \tag{7.6}$$

$$m \left| \lambda_{k,k} - \frac{\lambda_{k,-k}}{m-1} \right| + \sum_{k \neq i} \left| \lambda_{k,i} - \frac{\lambda_{k,-i}}{m-1} \right| - c \leq 0 \quad \text{for } i, k = 1, 2, \ldots, S \tag{7.7}$$

where $c \in (0, 1)$ is given.

The mid-layer update strategy using the discrete MCMC Model is outlined as follows. Consider a dynamically adjusting, human-intelligence-involved decision-making strategy for

updating an extended version of (5.8) by using the proposed discrete MCMC model (7.2) and the optimization problem (7.3). Since m $(= n)$ cues and S choices together for decision making, it would be more suitable to extend (5.8) to the case where d is a diagonal matrix instead of a scalar. Specifically, consider the following extension of (5.8):

$$W = (I_n - \mathbf{d})W^*_{\text{pro}} + \mathbf{d}W^*_{\text{con}} \tag{7.8}$$

where $\mathbf{d} = \text{diag}(d_{11}, \cdots, d_{nn}) \in \mathbb{R}^{n\times n}$ is a diagonal matrix. Further assume that the diagonal elements d_{ii} of $\mathbf{d}$ are dynamically adjusting and their values, can be chosen in the convex set $\text{conv}\{\mathrm{d}_i^{(1)}, \ldots, \mathrm{d}_i^{(S)}\}$ at every time instant $t = 0, 1, 2, \ldots$, where $\mathrm{d}_i^{(j)}$ is a scalar satisfying

$$\sum_{j=1}^{S} \mathrm{d}_i^{(j)} = 1, \quad \mathrm{d}_i^{(j)} \in [0, 1], \quad i = 1, \ldots, n \tag{7.9}$$

and

$$\text{conv}\{\mathrm{d}_i^{(1)}, \ldots, \mathrm{d}_i^{(S)}\} = \left\{ \sum_{j=1}^{S} \lambda_j \mathrm{d}_i^{(j)} : \lambda_j \in [0, 1], \sum_{j=1}^{S} \lambda_j = 1 \right\} \tag{7.10}$$

The probability of choosing $\mathrm{d}_i^{(j)}$ for d_{ii} at time instant t is denoted by $\mathrm{x}_i^{(j)}(t)$, i.e.,

$$p(d_{ii} = \mathrm{d}_i^{(j)}) = \mathrm{x}_i^{(j)}(t), \quad i = 1, \ldots, n, \quad j = 1, \ldots, S, \quad t = 0, 1, 2, \ldots$$

Define

$$\mathrm{x}^{(j)}(t) = \begin{bmatrix} \mathrm{x}_1^{(j)}(t) \\ \mathrm{x}_2^{(j)}(t) \\ \vdots \\ \mathrm{x}_n^{(j)}(t) \end{bmatrix}, \quad \mathbf{X}(t) = \begin{bmatrix} \mathrm{x}^{(1)}(t) \\ \mathrm{x}^{(2)}(t) \\ \vdots \\ \mathrm{x}^{(S)}(t) \end{bmatrix}$$

and we assume that the evolution of $\mathbf{X}(t)$ satisfies (7.2), i.e.,

$$\mathbf{X}(t+1) = \mathbf{H} \times \mathbf{X}(t) + \frac{1}{m-1}\mathbf{J} \times \mathbf{1} \equiv M_S \mathbf{X}(t) + \mathbf{b} \tag{7.11}$$

The final stationary distribution for $\mathbf{X}(t)$ is assumed to be known and given by

$$\begin{bmatrix} \hat{\mathbf{x}}^{(1)} \\ \hat{\mathbf{x}}^{(2)} \\ \vdots \\ \hat{\mathbf{x}}^{(S)} \end{bmatrix} = \lim_{t\to\infty} \mathbf{X}(t)$$

where $\hat{\mathbf{x}}^{(j)} = [\hat{\mathbf{x}}_1^{(j)}, \ldots, \hat{\mathbf{x}}_n^{(j)}]^{\mathrm{T}} \in \mathbb{R}^n$ satisfies

$$\sum_{j=1}^{S} \hat{\mathbf{x}}_i^{(j)} = 1, \quad \hat{\mathbf{x}}_i^{(j)} \in [0,1], \quad i = 1, \ldots, n \tag{7.12}$$

That is, the final probabilistic distribution of choosing $\mathrm{d}_i^{(j)}$ to exhibit some fixed pattern is optimal. For example, if more non-cooperative effect on $\mathbf{d}$ is desired, then the corresponding final probability will be greater than that of the cooperative choice. Now the proposed update strategy becomes solving the optimization problem (7.3) subject to (7.4)–(7.7).

Let

$$\tilde{\Lambda} = (\Lambda^+, \Lambda^-)$$

and

$$J_M(\tilde{\Lambda}) = \sum_{j=1}^{S} \sum_{k=1}^{S} \|b_{j,k} - \hat{\mathbf{x}}^{(j)}\|_1$$

Then (7.3) can be rewritten as

$$\min_{\tilde{\Lambda}} J_M(\tilde{\Lambda})$$

subject to

$$g(\tilde{\Lambda}) = 0$$

$$h(\tilde{\Lambda}) \leq 0$$

where $g(\cdot)$ denotes the left-hand side of (7.4) and (7.5), and $h(\cdot)$ denotes the left-hand side of (7.6) and (7.7).

To solve this optimization problem, we follow the similar bio-inspired, multiagent coordination idea to propose the following two-step algorithm for solving (7.3):

$$\begin{aligned}
\tilde{\Lambda}_{i,2k+1}^{(l)} &= \tilde{\Lambda}_{i,2k}^{(l)} + \alpha_k^{(l)} \Big[\arg \min_{\tilde{\Lambda}_i = \tilde{\Lambda}_{i,2k}^{(m)}, m \in N_c^{(l)} \cup \{l\}} J_M(\tilde{\Lambda}_i) - \tilde{\Lambda}_{i,2k}^{(l)} \Big] \\
&\quad + \delta_k^{(l)} \Big[\arg \min_{\tilde{\Lambda}_i = \tilde{\Lambda}_{i,2s}^{(m)}, m \in N_c^{(l)} \cup \{l\}, s = 0,1,\ldots,k} J_M(\tilde{\Lambda}_i) - \tilde{\Lambda}_{i,2k}^{(l)} \Big] \\
\tilde{\Lambda}_{i,2k+2}^{(l)} &= \arg \min_{\tilde{\Lambda}_i} \Big[\sum_{m \in N_c^{(l)} \cup \{l\}} \|\tilde{\Lambda}_i - \tilde{\Lambda}_{i,2k+1}^{(m)}\|_{\mathrm{F}}^2 \Big], \\
& \qquad i = 1, \ldots, n, \quad l = 1, \ldots, M, \quad k = 0, 1, 2, \ldots
\end{aligned}$$

When updating $\tilde{\Lambda}_i$ using the first step algorithm, one needs to check its compatibility with the constraints (7.4)–(7.7). Specifically, restrict $\lambda_{j,\pm k} \in [0,1]$, $j,k = 1,\ldots,S$, $k \neq j$, so that $\sum_{k=1,k\neq j}^{S} \lambda_{j,\pm k} < 1$ for every $j = 1,\ldots,S$. Then let $\lambda_{j,\pm j} = 1 - \sum_{k=1,k\neq j}^{S} \lambda_{j,\pm k}$ for every $j = 1,\ldots,S$. Such choice of $\lambda_{j,\pm k}$, $j,k = 1,\ldots,S$, will satisfy (7.4)–(7.6). Next, put this choice into the left-hand side of (7.7) to see if the inequality is satisfied. If yes, then continue evaluating the cost function and updating $\tilde{\Lambda}$. If not, then this selection process is repeated until a compatible choice for (7.4)–(7.7) is found.

Once the MCMC model (7.11) is determined by the above optimization problem, dynamically update W as follows

$$W = (I_n - \mathbf{d})W^*_{\text{pro}} + \mathbf{d}W^*_{\text{con}}, \quad t = 0, 1, 2, \ldots \tag{7.13}$$

where $\mathbf{d} = \text{diag}(d_{11}, \cdots, d_{nn})$ is updated by using either the weighted additive (WADD) strategy

$$d_{ii} = \sum_{j=1}^{S} \mathrm{d}_i^{(j)} \mathrm{x}_i^{(j)}(t), \quad i = 1, \ldots, n \tag{7.14}$$

so that $d_{ii} \in \text{conv}\{\mathrm{d}_i^{(1)}, \ldots, \mathrm{d}_i^{(S)}\}$, or the take-the-best (TTB) strategy

$$d_{ii} = \mathrm{d}_i^{(s)}, \quad s \in \left\{ m : \mathrm{x}_i^{(m)}(t) = \max_{1 \le j \le S} \{\mathrm{x}_i^{(j)}(t)\} \right\} \tag{7.15}$$

so that $d_{ii} \in \{\mathrm{d}_i^{(1)}, \ldots, \mathrm{d}_i^{(S)}\} \subset \text{conv}\{\mathrm{d}_i^{(1)}, \ldots, \mathrm{d}_i^{(S)}\}$. The WADD strategy mimics the human decision making under the normal situation while the TTB strategy mimics the human decision making under the emergency situation. Depending on the extent of nuclear escalation, $\mathbf{d}$ could be updated by the WADD at steady-state times, while the TBB strategy may take over it when the situation becomes tense.

The SEM in (2.7) is a fixed-structure model whose structural graph topology between different factors is given. Such a model becomes problematic when the correlation between these factors is either unclear, a priori, or dynamically changing. One possible scenario for this situation is that the nuclear deployment needs to be adjusted as the nuclear competing race intensifies or weakens. Some strongly correlated factors may become less dependent now, or vice versa. In this case, the graph topology C in (2.7) is not given or the weight matrix A is not fixed anymore. Previously it was assumed that C was given for (2.7). Now this assumption is dropped by proposing a method to identify possible topology matrix C and calculate weight matrix A simultaneously by using the input, the output, and the

measurement or estimation data for (2.7).

Specifically, consider the model

$$z_i = \sum_{j=1, j\neq i}^{n} d_{ij}\tilde{x}_j + u_i, \quad i = 1, \ldots, n \tag{7.16}$$

$$\tilde{x}_j = x_j + \Delta x_j \tag{7.17}$$

$$\tilde{y}_i = z_i + \Delta y_i \tag{7.18}$$

where there is no topological constraint. Instead, the idea is to identify possible graph topology for this model by determining d_{ij} through the input data u_i, the data received and/or estimated from all other nodes $\tilde{x}_j$, and the output data $\tilde{y}_i$.

Identifying possible $C = [c_{ij}]_{i,j=1,\ldots,n}$, $c_{ii} = 0$, is straightforward: If $d_{ij} \neq 0$, $i, j = 1, \ldots, n$, $i \neq j$, then $c_{ij} = 1$. Otherwise, if $d_{ij} = 0$, $i, j = 1, \ldots, n$, $i \neq j$, then $c_{ij} = 0$. That is, if the identified weight between two nodes of a graph is nonzero, then it is highly possible that there is a link between these two nodes. Otherwise, there may not be a reliable link between these two nodes. Of course, there is a small chance that while the real topology is one graph, the generated graph induced by d_{ij} may not be the same due to some numerical errors on d_{ij}. Hence, to validate whether this idea would generate a reasonable topology for (7.16), one needs to define an appropriate validation metric.

We consider how a perturbation in the model (7.16)–(7.18) will affect the generated C. An important observation is that if the generated graph C is a reliable one, then the small perturbation for $\tilde{x}_j$ and $\tilde{y}_i$ in (7.16)–(7.18) should not lead to a major change of its topological structure. Let $D = [d_{ij}]_{i,j=1,\ldots,n}$, $d_{ii} = 0$, be an identified matrix for (7.16)–(7.18) through $\tilde{x}_j$, $\tilde{y}_i$, and u_i. Moreover, let $\hat{D} = [\hat{d}_{ij}]_{i,j=1,\ldots,n}$, $\hat{d}_{ii} = 0$, be an identified

matrix for the perturbed model

$$z_i = \sum_{j=1, j\neq i}^{n} \hat{d}_{ij}\hat{x}_j + u_i, \quad i = 1, \ldots, n \tag{7.19}$$

$$\hat{x}_j = \tilde{x}_j + \delta\tilde{x}_j \tag{7.20}$$

$$\hat{y}_i = \tilde{y}_i + \delta\tilde{y}_i \tag{7.21}$$

$$\tilde{x}_j = x_j + \Delta x_j \tag{7.22}$$

$$\tilde{y}_i = z_i + \Delta y_i \tag{7.23}$$

through $\hat{x}_j$, $\hat{y}_i$, and u_i, where $\delta\tilde{x}_j$ and $\delta\tilde{y}_i$ denote the small variation of $\tilde{x}_j$ and $\tilde{y}_i$, respectively. Finally, let C be the graph topology generated by D and $\hat{C}$ be the graph topology generated by $\hat{D}$ through the above proposed rule. Then the rigorous definition of the validation metric M_v is given by

$$M_v = \lim_{(\delta\tilde{x}, \delta\tilde{y}) \to (0,0)} \frac{\|\hat{C} - C\|}{\|(\delta\tilde{x}, \delta\tilde{y})\|} \tag{7.24}$$

where $\delta\tilde{x} = (\delta\tilde{x}_1, \ldots, \delta\tilde{x}_n)$ and $\delta\tilde{y} = (\delta\tilde{y}_1, \ldots, \delta\tilde{y}_n)$. Clearly $M_v \geq 0$ and the smaller value M_V can achieve, the better result C can generate. Ideally for $M_v = 0$, such a generated C should be a reliable one. Practically if M_v does not exceed some small threshold, i.e., $M_v \leq \varepsilon << 1$, then we think such a generated C is a reasonable estimate. Also in practice, it is hard to evaluate the limit in this definition, often one can take an approximate form

$$M_v \approx \frac{\|\hat{C} - C\|}{\|(\delta\tilde{x}, \delta\tilde{y})\|} \tag{7.25}$$

for some small $\delta\tilde{x}$ and $\delta\tilde{y}$.

The method of identifying D and $\hat{D}$ will be similar to that of identifying W in (2.9)–(2.11) by using the proposed cooperative game and non-cooperative game approaches.

Specifically, D is updated by

$$D = (I_n - \mathbf{d})D^*_{\text{pro}} + \mathbf{d}D^*_{\text{con}} \tag{7.26}$$

where $\mathbf{d}$ is updated by the proposed WADD or TTB strategy, D^*_{pro} is the result of the cooperative game approach for the model (7.16)–(7.18), and D^*_{con} is the result of the non-cooperative game approach for the model (7.16)–(7.18). Similarly, for $\hat{D} = (I_n - \mathbf{d})\hat{D}^*_{\text{pro}} + \mathbf{d}\hat{D}^*_{\text{con}}$, $\mathbf{d}$ is updated by the proposed WADD or TTB strategy, $\hat{D}^*_{\text{pro}}$ is the result of the cooperative game approach for the model (7.19)–(7.23), and $\hat{D}^*_{\text{con}}$ is the result of the non-cooperative game approach for the model (7.19)–(7.23). Once we obtain $D = [d_{ij}]_{i,j=1,\ldots,n}$, $d_{ii} = 0$, and $\hat{D} = [\hat{d}_{ij}]_{i,j=1,\ldots,n}$, $\hat{d}_{ii} = 0$, $C = [c_{ij}]_{i,j=1,\ldots,n}$ and $\hat{C} = [\hat{c}_{ij}]_{i,j=1,\ldots,n}$ can be obtained as follows:

$$c_{ij} = \begin{cases} 1, & d_{ij} \neq 0 \\ 0, & d_{ij} = 0 \end{cases}, \quad \hat{c}_{ij} = \begin{cases} 1, & \hat{d}_{ij} \neq 0 \\ 0, & \hat{d}_{ij} = 0 \end{cases}$$

Chapter 8

Simulations of Multi-cue Multi-choice Tasks

In this section, the preliminary simulation details and results for the proposed human decision making dynamics are outlined. The algorithm formulated above, Modeling Human decision-making dynamics via multi-cue multi-choice tasks is simulated to show its effectiveness. The software MATLAB R2021b is used to apply and test the proposed approaches by means of randomly generated data.

To dynamically adjust the decision making strategy, simulations were done by implementing the proposed discrete MCMC (7.2) and the optimization problem (7.3). The implementation of this model is as follows. First, input parameters were set for the discrete time 2AFC model. The Markov process is over the time interval $[0, t]$ and divided into sub-intervals of length τ. The size each sub-interval is $\pm\Delta$, and the state space is given by S. Then, k is number of time steps in the process and m is the size of the state space. These inputs can be seen in Table 8.1.

Now, the discrete time 2AFC model is applied to the discrete time MCMC model. The matrix form of this model is outlined in Equation (7.2). Here, H and j are formulated given a random λ matrix and P, where

$$P^{(ij)} = \begin{cases} \frac{1}{m} & \text{if } j = i \\ 0 & \text{otherwise} \end{cases}$$

Table 8.1: *Parameter Inputs to the Proposed Algorithm*

Parameter Name	Parameter Value
t	1
τ	1
k	t/τ
Δ	$\sqrt{t}$
S	$[-1, 0, 1]$
m	$2k + 1$

Next, the discrete time MCMC model was optimized using the two step bio-inspired, multiagent coordination algorithm, that is similar to Section 6. The initial parameters for the optimization can be seen in Table 8.2. Here, M is the number of agents, and α and δ are the learning rates. Some initial parameters for Equation (7.3) are also outline in Table 8.2. Here, $\mathbf{x}^{(k)}$ and $\mathbf{x}^{(j)}$ are the state probability distribution vectors. With this information, the MCMC model was able to be optimized.

Table 8.2: *Parameter Inputs to the Proposed Algorithm*

Parameter Name	Parameter Value
M	4
α	.01
δ	.01
$x^{(k)}$	$[0, 1, 1]$
$x^{(j)}$	$[0, 1, 1]$

Following the optimization, $\mathbf{d}$ is updated using the WADD strategy. Given a $d^{(j)}$ of $1/3$, d_{ii} was found and can be seen in Equation (8.1). To achieve a $\mathbf{d}$ vector in the set $[0, 1]$, d_{ii} is normalized using the softmax function. The final $\mathbf{d}$ can be seen in Equation (8.2).

$$d_{ii} = \begin{pmatrix} 1.0909 \\ 1.3645 \\ 1.0465 \end{pmatrix} \tag{8.1}$$

$$\mathbf{d} = \begin{pmatrix} 0.3057 \\ 0.4019 \\ 0.2924 \end{pmatrix} \tag{8.2}$$

Chapter 9

Modeling the intertwined dynamics of the top network layer, the middle network layer, and bottom network layer to the hybrid game model

The hybrid game model or bottom level characterizes the interconnected dynamics that bring the top layer model, the middle layer model and the bottom layer model together to function as a unified, coupled spatialtemporal process featuring intrinsic escalation dynamics by means of AI, acyclic graphs, and a compartmental network approach.

The mathematical modeling of the top level consists of the information retrieving network (RN), information analyzing network (AN), and information formulating network (FN) for processing input data. An input-output cellular neural network (CNN) model is proposed for RN, a Bayesian belief network (BBN) model is proposed for AN, and a deep neural network (DNN) model is proposed for FN.

The purpose of RN is to convert non-visual spatial information of data into geometric maps, which can facilitate the generation of the ultimate visual map for escalation competing dynamics. This job can be done by utilizing a CNN. In this thesis, first-order cell dynamics and linear interactions are used. The state equation of a cell in position (i, j) is given by

the following nonlinear differential equation

$$\frac{dz_{ij}(t)}{dt} = -z_{ij}(t) + \sum_{(k,l)\in N(i,j)} A(i,j;k,l)\cdot y_{kl}(t) + \sum_{(k,l)\in N(i,j)} B(i,j;k,l)\cdot u_{kl}(t) + v(i,j;k,l) \tag{9.1}$$

$$y_{ij}(t) = \frac{1}{2}\Big(|z_{ij}(t)+a| - |z_{ij}(t)-a|\Big) \tag{9.2}$$

where $u_{ij}(\cdot)$, $z_{ij}(t)$, and $y_{ij}(\cdot)$ are the input, the state, and the output of the cell in position (i,j), respectively; the indices k and l denote a generic cell belonging to the neighborhood $N(i,j)$ of the cell in position (i,j), $v(i,j;k,l)$ denotes the bias of CNN for position (i,j) associated with the neighboring cell (k,l), and $a > 0$ denotes the value of the state at which the output hits the (upper) threshold.

In this project, the models (9.1) and (9.2) are used to describe the data processing dynamics of raw spatial data before feeding it into the SEM (2.6). The output of (9.1) and (9.2) serves as part of the input data to the SEM (2.6).

The purpose of AN is to predict temporal evolution of data and generate a synthetic counterpart. The most common approaches for doing this, such as Kalman filtering and particle filtering, can all be unified under the same framework–BNNs. The standard construction of a BBN assumes prior expert knowledge of the underlying domain. The first step is to build a directed acyclic graph, followed by the second step to assess the conditional probability distribution in each node. Specifically, let $G(\mathbf{X}, E)$ denote an annotated directed acyclic graph, where the nodes are random variables $X_i \in \mathbf{X}$. Furthermore, let $\theta_i = p(X_i \,|\, \text{Ancestors}(X_i))$ be the probabilistic conditional distributions defined for each X_i. Then a Bayesian belief network uniquely specifies a joint distribution given by

$$p(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} p(X_i \,|\, \text{Parents}(X_i))$$

The joint probability can be expanded by the Bayes chain rule as follows under the Markov

assumption (each variable is independent of its non-descendents, given its parents)

$$p(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} p(X_i \,|\, X_1, X_2, \ldots, X_{i-1})$$

While the probabilistic framework for BBN is intuitive, it is hard to express this formulation into a compact, computational form like (9.1) to unify CNN, BBN, and DNN together. Reference [4] developed a new approach for inference in BNN by using the idea of partial differentiation for multivariate polynomial functions. It is motivated by the analogue of chain rules under partial differentiation operation and the Bayes law. Moreover, it presents a mathematical model for dynamic analysis of the AN layer and 2D/3D map generation for visualization.

To elaborate this model, variables are denoted by upper-case letters (A) and their values by lower-case letters (a). Sets of variables are denoted by bold-face upper-case letters ($\mathbf{A}$) and their instantiations are denoted by bold-face lower-case letters ($\mathbf{a}$). Next, let $\mathcal{F} = \{X\} \cup \mathbf{U}$ be the family of variable X and let $\mathbf{f} = x\mathbf{u}$ be a corresponding instantiation. Then use $\theta_{\mathbf{f}}$ and/or $\theta_{x\mathbf{u}}$ to represent the conditional probability $p(x \,|\, \mathbf{u})$. Moreover, $\mathbf{x} \sim \mathbf{f}$ will mean that instantiations $\mathbf{x}$ and $\mathbf{f}$ are consistent.

Definition 9.0.1 ([4])**.** *Let $\mathcal{N}$ be a BNN with variables $\mathbf{X} = X_1, \ldots, X_n$ and families $\mathbf{F}_1, \ldots, \mathbf{F}_n$. Then*

$$\mathcal{F}(\lambda_{x_i}, \theta_{\mathbf{f}_i}) = \sum_{\mathbf{x}} \prod_{\mathbf{f}_i \sim \mathbf{x}} \theta_{\mathbf{f}_i} \prod_{x_i \sim \mathbf{x}} \lambda_{x_i} \tag{9.3}$$

is called the canonical polynomial of BNN $\mathcal{N}$, where λ_{x_i} are called evidence indicators and $\theta_{\mathbf{f}_i}$ are called network parameters. A quantification Θ of a BNN is a function that assigns a value $\Theta(\mathbf{f})$ to each instantiation $\mathbf{f}$ of family $\mathbf{F}$. The value of indicator λ_x at instantiation $\mathbf{e}$, denoted $\mathbf{e}(x)$, is 1 if x is consistent with $\mathbf{e}$, and is 0 otherwise. The value of polynomial

$\mathcal{F}$ under evidence $\mathbf{e}$ *and quantification* Θ *is defined as*

$$\mathcal{F}(\mathbf{e}, \Theta) := \mathcal{F}(\lambda_{x_i} = \mathbf{e}(x_i), \theta_{\mathbf{f}_i} = \Theta(\mathbf{f}_i))$$

Let $\mathbf{e}$ be an instantiation of evidence, $\mathbf{E}$ be a set of evidences, and $\mathbf{X}$ be a set of variables. Then $\mathbf{e} - \mathbf{X}$ denotes the subset of instantiation $\mathbf{e}$ pertaining to variables not appearing in $\mathbf{X}$. For each instantiation $\mathbf{e}$ and quantification Θ, the polynomial (9.3) can be evaluated to compute the probability of $\mathbf{e}$ for given Θ. Often $\mathcal{F}(\mathbf{e})$ will be written instead of $\mathcal{F}(\mathbf{e}, \Theta)$ when no ambiguity is anticipated.

The polynomial framework (9.3) possesses the following key properties for BNNs, which show some similarity between calculus operation and conditional probability.

Theorem 9.0.1 ([4])**.** *Let $\mathcal{N}$ be a BNN representing probability distribution $p(\cdot)$ and having canonical polynomial $\mathcal{F}$. Then the following statements hold:*

1) For every evidence $\mathbf{e}$ *and quantification* Θ*,*

$$\mathcal{F}(\mathbf{e}, \Theta) = p(\mathbf{e} \mid \Theta) \tag{9.4}$$

$$\mathcal{F}(\mathbf{e}) = p(\mathbf{e}) \tag{9.5}$$

2) For every variable X, family $\mathbf{F}$, and evidence $\mathbf{e}$*,*

$$\frac{\partial \mathcal{F}(\mathbf{e}, \Theta)}{\partial \lambda_x} = p(x, \mathbf{e} - X \mid \Theta) \tag{9.6}$$

$$\frac{\partial \mathcal{F}(\mathbf{e}, \Theta)}{\partial \theta_{\mathbf{f}}} = \frac{p(\mathbf{f}, \mathbf{e} \mid \Theta)}{\Theta(\mathbf{f})} \tag{9.7}$$

3) For every pair of variables $X \neq Y$, pair of families $\mathbf{F}_1 \neq \mathbf{F}_2$, and evidence $\mathbf{e}$,

$$\frac{\partial^2 \mathcal{F}(\mathbf{e}, \Theta)}{\partial \lambda_x \partial \lambda_y} = p(x, y, \mathbf{e} - XY \,|\, \Theta) \tag{9.8}$$

$$\frac{\partial^2 \mathcal{F}(\mathbf{e}, \Theta)}{\partial \lambda_x \partial \theta_{\mathbf{f}_1}} = \frac{p(x, \mathbf{f}_1, \mathbf{e} - X \,|\, \Theta)}{\Theta(\mathbf{f}_1)} \tag{9.9}$$

$$\frac{\partial^2 \mathcal{F}(\mathbf{e}, \Theta)}{\partial \theta_{\mathbf{f}_1} \partial \theta_{\mathbf{f}_2}} = \frac{p(\mathbf{f}_1, \mathbf{f}_2, \mathbf{e} \,|\, \Theta)}{\Theta(\mathbf{f}_1)\Theta(\mathbf{f}_2)} \tag{9.10}$$

4) For $X \notin \mathbf{E}$,

$$p(x \,|\, \mathbf{e}) = \frac{\frac{\partial \mathcal{F}(\mathbf{e})}{\partial \lambda_x}}{\mathcal{F}(\mathbf{e})} \tag{9.11}$$

5) For every variable X, family $\mathbf{F}$, and evidence $\mathbf{e}$,

$$p(\mathbf{e} - X) = \sum_x \frac{\partial \mathcal{F}(\mathbf{e})}{\partial \lambda_x} \tag{9.12}$$

$$p(x \,|\, \mathbf{e} - X) = \frac{\frac{\partial \mathcal{F}(\mathbf{e})}{\partial \lambda_x}}{\sum_x \frac{\partial \mathcal{F}(\mathbf{e})}{\partial \lambda_x}} \tag{9.13}$$

$$p(\mathbf{f} \,|\, \mathbf{e}, \Theta) = \frac{\frac{\partial \mathcal{F}(\mathbf{e}, \Theta)}{\partial \theta_{\mathbf{f}}}}{\mathcal{F}(\mathbf{e}, \Theta)} \Theta(\mathbf{f}) \tag{9.14}$$

Next, the model (9.3) is extended to the dynamic case where it involves temporal evolution of variables. Specifically, two kinds of continuous-time Markov processes are condsidered for each variable $X_i = X_i(t)$, i.e., jump processes and diffusion processes. Jump processes assume that in a small time interval there is an overwhelming probability that the state will remain unchanged; however, if it changes, the change may be radical. Diffusion processes are represented by diffusion and by Brownian motion; there it is certain that some change will occur in any time interval, however small; only, here it is certain that the changes during small time intervals will be also small.

For Markov jump processes, the state space Ω_i for each variable X_i is (finitely or infinitely) countable. In this case, the time evolution of variable $X_i = X_i(t) \in \Omega_i$ can be

described by the Kolmogorov forward and backward equations as follows

$$\frac{\partial p_{xy}(s;t)}{\partial t} = \sum_{z\in\Omega_i} p_{xz}(s;t)A_{zy}(t), \quad x,y\in\Omega_i, \quad t>s\geq 0, \quad i=1,\ldots,n \tag{9.15}$$

$$\frac{\partial p_{xy}(s;t)}{\partial s} = -\sum_{z\in\Omega_i} A_{xz}(s)p_{zy}(s;t), \quad x,y\in\Omega_i \tag{9.16}$$

where $p_{xy}(s;t) = p(x,s;y,t)$ denotes the probability that the system in state $x\in\Omega_i$ at time s jumps to state $y\in\Omega_i$ at later time $t>s$, $t>s\geq 0$ are the final and initial times, respectively, and $A_i(t) = [A_{xy}(t)]_{x,y=\in\Omega_i}$ is the transition rate matrix (also known as the generator matrix) for variable X_i, which satisfies

$$A_{xy}(t) = \frac{\partial p_{xy}}{\partial u}(t;u)\Big|_{u=t}, \quad A_{yz}(t)\geq 0,\ y\neq z,\ y,z\in\Omega_i, \quad \sum_{z\in\Omega_i} A_{yz}(t) = 0$$

For diffusion processes, the state space Ω_i for each variable X_i is a continuum. In this case, the time evolution of variable $X_i = X_i(t)\in\Omega_i$ can be described by the Kolmogorov equations as well. If we assume the variable $X_i(t)$ evolves according to the stochastic differentiable equation

$$dX_i(t) = \mu(X_i(t),t)dt + \sigma(X_i(t),t)dW(t), \quad t\geq s \tag{9.17}$$

then

$$\frac{\partial}{\partial t}p(x,t) = -\frac{\partial}{\partial x}[\mu(x,t)p(x,t)] + \frac{1}{2}\frac{\partial^2}{\partial x^2}[\sigma^2(x,t)p(x,t)] \tag{9.18}$$

for $t\geq s$, with the initial condition $p(x,s) = p_s(x)$; and

$$-\frac{\partial}{\partial s}p(x,s) = \mu(x,s)\frac{\partial}{\partial x}p(x,s) + \frac{1}{2}\sigma^2(x,s)\frac{\partial^2}{\partial x^2}p(x,s) \tag{9.19}$$

for $s\leq t$, with the final condition $p(x,t) = p_t(x)$.

Informally, the Kolmogorov forward equation addresses the following problem. There is information about the state x of the system at time s (namely a probability distribution $p_s(x)$); we want to know the probability distribution of the state at a later time $t > s$. A similar remarks holds for the Kolmogorov back equation.

These Markov processes described by the Kolmogorov equations are applied to the polynomial framework of BNNs. Specifically, it follows from the Bayes law, (9.5), and (9.11) that

$$p(x) = p(x \,|\, \mathbf{e})p(\mathbf{e}) = \frac{\frac{\partial \mathcal{F}(\mathbf{e})}{\partial \lambda_x}}{\mathcal{F}(\mathbf{e})}\mathcal{F}(\mathbf{e}) = \frac{\partial \mathcal{F}(\mathbf{e})}{\partial \lambda_x} \tag{9.20}$$

If $p(x)$ involves a time evolution process, i.e., $p(x) = p(x,t)$, depending on whether it is a jump process or a diffusion process, either (9.15) or (9.18) is satisfied in forward time, or, either (9.16) or (9.19) is satisfied in backward time. In this project, two forward time processes are taken for a case study. Thus, for the jump process in forward time, the dynamic model for BNNs is given by

$$\frac{\partial^3 \mathcal{F}(\mathbf{e}, s; t)}{\partial \lambda_x \partial \lambda_y \partial t} = \sum_{z \in \Omega_i} \frac{\partial^2 \mathcal{F}(\mathbf{e}, s; t)}{\partial \lambda_x \partial \lambda_z} A_{zy}(t), \quad x, y \in \Omega_i, \quad t > s \geq 0, \quad i = 1, \ldots, n \tag{9.21}$$

For the diffusion process, the dynamic model for BNNs is given by

$$\frac{\partial^2 \mathcal{F}(\mathbf{e}, t)}{\partial \lambda_x \partial t} = -\frac{\partial}{\partial x}\Big[\mu(x,t)\frac{\partial \mathcal{F}(\mathbf{e}, t)}{\partial \lambda_x}\Big] + \frac{1}{2}\frac{\partial^2}{\partial x^2}\Big[\sigma^2(x,t)\frac{\partial \mathcal{F}(\mathbf{e}, t)}{\partial \lambda_x}\Big], \quad t \geq s \tag{9.22}$$

with the initial condition $p(x, s) = \frac{\partial \mathcal{F}(\mathbf{e}, s)}{\partial \lambda_x}$. In the simulation verification, the diffusion process will be considered and $\mu(x,t) = \mu > 0$ and $\sigma(x,t) = \sigma > 0$ will be assumed for simplicity. Obviously (9.22) is a partial differential equation. One needs to develop numerical approaches such as finite elements to solve it. The output of this equation gives numerical values of $\frac{\partial \mathcal{F}(\mathbf{e}, t)}{\partial \lambda_x}$ and $\mathcal{F}(\mathbf{e}, t)$, which are the same as $p(x,t)$ and $p(\mathbf{e}, t)$, respectively. They are used to calculate the expectation of variables X_i under evidence $\mathbf{e}$. The result

is a time-series prediction of synthetic data serving as the input dataset for the non-fixed-structure SEM.

The purpose of FN is to transform data from the existing domain to another domain to better visualize its hidden feature by conducting discrete convolution and averaging operation. This can be done through an ℓ-layer, feed-forward DNN, which is given by the form

$$w_0 = u \tag{9.23}$$

$$w_i = \Phi_i(W_i w_{i-1} + b_i) \tag{9.24}$$

$$y = W_{\ell+1} w_\ell + b_{\ell+1} \tag{9.25}$$

where $u \in \mathbb{R}^{n_0}$ denotes the input to DNN, $w_i \in \mathbb{R}^{n_i}$ denote the outputs from the ith layer of DNN, $i = 1, 2, \ldots, \ell$, and $y \in \mathbb{R}^{n_{\ell+1}}$ denotes the final output from DNN. The operations for each layer of DNN are defined by a weight matrix $W_i \in \mathbb{R}^{n_i \times n_{i-1}}$, bias vector $b_i \in \mathbb{R}^{n_i}$, and activation function $\Phi_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$. The activation function $\Phi_i(\cdot)$ is applied elementwise, that is, $\Phi_i(v) = [\phi(v_1), \ldots, \phi(v_{n_i})]^{\mathrm{T}}$, where $\phi : \mathbb{R} \to \mathbb{R}$ is a selected scalar activation function, e.g., ReLU $\phi(v) = \max(0, v)$, sigmoid $\phi(v) = 1/(1 + e^{-v})$, hyperbolic tangent $\phi(v) = \tanh(v)$, etc.

Note that (9.23)–(9.25) is a set of equations implemented at the same time step, that is, they should be understood as

$$w_0(t) = u(t) \tag{9.26}$$

$$w_i(t) = \Phi_i(W_i w_{i-1}(t) + b_i) \tag{9.27}$$

$$y(t) = W_{\ell+1} w_\ell(t) + b_{\ell+1} \tag{9.28}$$

where $t \geq 0$. Depending on the application, ℓ for a DNN could vary from one to a hundred. A similar remark holds for n_i. In this thesis, we take $\ell = 3$ and $n_i = n = 5$ as an example to simulate the whole process, $i = 0, 1, 2, 3, 4$. The output of this DNN model serves as part

of the input data to the SEM (2.6).

Chapter 10

Implementation of BBN

Bayes Theorem (BT) provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself. Here, $P(h)$ is the initial probability of the hypothesis h and $P(D)$ is the prior probability that data D will be observed. Next, the probability $P(h|D)$ that h holds given the observed training data D. $P(h|D)$ is called the posterior probability of h, because it reflects the confidence that h holds seeing the training data D [18]. Bayes theorem is the cornerstone of Bayesian learning methods because it provides a way to calculate the posterior probability, from the prior probability $P(h)$, together with $P(D)$ and $P(D|h)$. Bayes Theorem is outlined below:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)} \tag{10.1}$$

Consider a set of Hypotheses H, where a maximum a posteriori (MAP) is finding the most probable hypothesis $h \in H$ given the observed data, D. Then the MAP can be determined using Equation 10.1 as follows:

$$\begin{aligned} h_{MAP} &= \arg\min_{h \in H} P(h|D) \\ &= \arg\min_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \arg\min_{h \in H} P(D|h)P(h) \end{aligned} \tag{10.2}$$

In this case, it is assumed that every hypothesis in H is an equally probable priori. For this case, Equation 10.2 can be simplified to find the most probable hypothesis. Here, $P(D|h)$ is the likelihood of data, D given h. Thus, any hypothesis that maximizes the likelihood will be the maximum likelihood (ML) as shown below:

$$h_{ML} = \arg\min_{h \in H} P(D|h) \tag{10.3}$$

BT is then be used to formulate a BBN. A BBN describes the probability distribution governing a set of variables by specifying a set of conditional independence assumptions along with a set of conditional probabilities. In general, a Bayesian belief network represents the joint probability distribution by specifying a set of conditional independence assumptions (represented by a DAG), together with sets of local conditional probabilities. Each variable in the joint space is represented by a node in the Bayesian network. For each variable two types of information are specified. First, the network arcs represent the assertion that the variable is conditionally independent of its nondescendants in the network given its immediate predecessors in the network. X is a descendant of, Y, if there is a directed path from Y to X. Second, a conditional probability table is given for each variable, describing the probability distribution for that variable given the values of its immediate predecessors [18]. Then a BBN uniquely specifies a joint distribution.

A BBN can also be used for inference given observed values of other variables. Inference is used to find the probability distribution for a target variable, which specifies the probability that it will take on each of its possible values given the observed values of the other variables. However, before this is possible the BBN must be trained.

The EM algorithm is used in training the BBN. The EM algorithm is used to learn in the presence of unobserved variables. The EM algorithm can be used even for variables whose value is never directly observed, provided the general form of the probability distribution governing these variables is known. The maximum likelihood hypothesis minimizes the sum of squared errors over m training instances. Here, D is a set of instances generated by a

probability distribution that is a mixture of k distinct normal distributions. One of the k distributions is selected, and a single random instance x_i, the sum of squared error is minimized by the sample mean [18]. This process is repeated to generate a set of data and can be seen in the equation below:

$$\mu_{ML} = \arg\min_{\mu} \sum_{i=1}^{m} (x_i - \mu)^2 \tag{10.4}$$

Continuing with the EM algorithm, assume x_i is the observed variable in the description, and z_{i1} and z_{i2} indicate the normal distribution that was used to generate the value x_i. Here, z_{ij} has the value 1 if x_i was created by the jth normal distribution. Otherwise, it will have a value of 0. Here, x_i is the observed variable and z_{i1} and z_{i2} are hidden variables. If the values of z_{il} and z_{i2} were observed, Equation 10.4 could be used to solve for the means μ_1 and μ_2. However, this is not the case so the EM algorithm is used. The EM algorithm searches for a maximum likelihood hypothesis by repeatedly re-estimating the expected values of the hidden variables given its current hypothesis. The ML hypothesis is then re-calculated using the expected values for the hidden variables. The EM two step process is outlined next. First the expected value of z_{ij} is calculated:

$$E[z_{ij}] = \frac{P(x = x_i | \mu = \mu_j)}{\sum_{n=1}^{2} P(x = x_i | \mu = \mu_n} \tag{10.5}$$

In the next step, $E[z_{ij}]$ is used to derive the ML hypothesis $h' = (\mu'_1, \mu'_2)$. The ML hypothesis can be seen below:

$$\mu_j = \frac{\sum_{i=1}^{m} E[z_{ij}] x_i}{\sum_{i=1}^{m} E[z_{ij}]} \tag{10.6}$$

A Gaussian Mixture Model (GMM) assumes the data to be segregated into clusters in such a way that each data point in a given cluster follows a particular Multi-variate Gaussian distribution and the Multi-Variate Gaussian distributions of each cluster is independent of one another. To cluster data in such a model, the posterior probability of a data-point belonging to a given cluster given the observed data needs to be calculated. A method to

perform this is BBNs.

A GMM was implemented using a BBN to generate synthetic data. The purpose of the GMM model is to classify countries into distinct categories. Once classified, synthetic data can be created based on these classified countries.

The data used in this problem can be found on World Bank Open's website [1]. To train the model, three different training inputs were used:

- Countries Gross Domestic Product (GDP)
- Countries Population (POP)
- Countries Military Expenditures (ME)

Data for these training inputs came from 183 countries spanning 11 years from 2009 to 2020. The data is split into 4 different classes according to the GDP of that country. Since this model is proof of concept, splitting the countries into classes is arbitrary. When real world data is implemented using this model, classifying each county will be done by an expert with extensive knowledge.

The model of the GMM will be outlined next. The structure of the model can be seen in Figure 10.1. Here, node 1 is class for each country. Nodes 2, 3, and 4 are the discrete components that can be used to classify nodes 5, 6, and 7. Nodes 5, 6, and 7 are Gaussian values that correspond to GDP, POP, and ME. This model is referred to as a conditional Gaussian model because there are directed edges from the discrete nodes (2, 3, 4) to the continuous nodes (5, 6, 7).

The model then was trained using the EM algorithm outlined earlier in this section. Once trained, the model is able to generate synthetic data from the conditional probabilities of each class.

The results from the GMM model are outlined below. Figure 10.2 has 2 plots. The top plot shows the scaled training data that was used to train the model. The bottom plot show

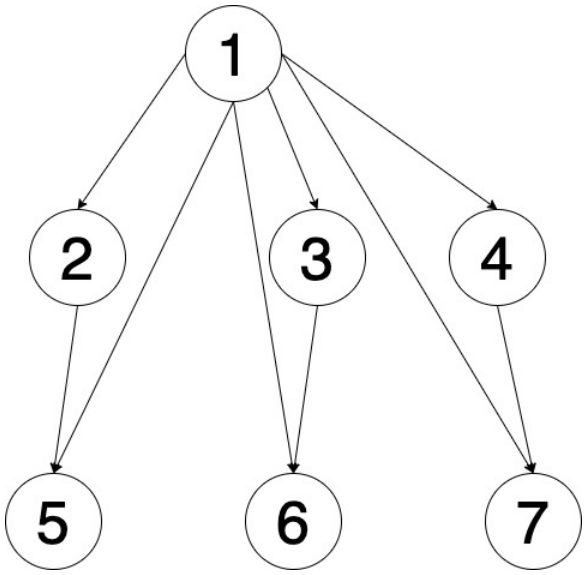

Figure 10.1: DAG of Model

the synthetic data from the model that was output when the training data was sampled using the trained GMM.

Figure 10.3 shows the the model when the testing data set was run through it. The top plot is the test data that was fed into the GMM and the bottom plot shows the synthetic data that was output from the GMM.

Figure 10.4 gives the probability distribution of each class from the test data,

$$P(\text{class}|\text{evidence})$$

This probability distribution gives the probability that each country will be in a certain class given the test data as evidence.

Figure 10.5 show the synthetic from the GMM. The top plot shows synthetic data without any Gaussian noise added to it. The bottom plot shows synthetic data with Gaussian noise added to it. In this way, synthetic data can be generated to create synthetic countries. These synthetic countries can then be fed into the bottom layer and used to verify the escalation dynamics model.

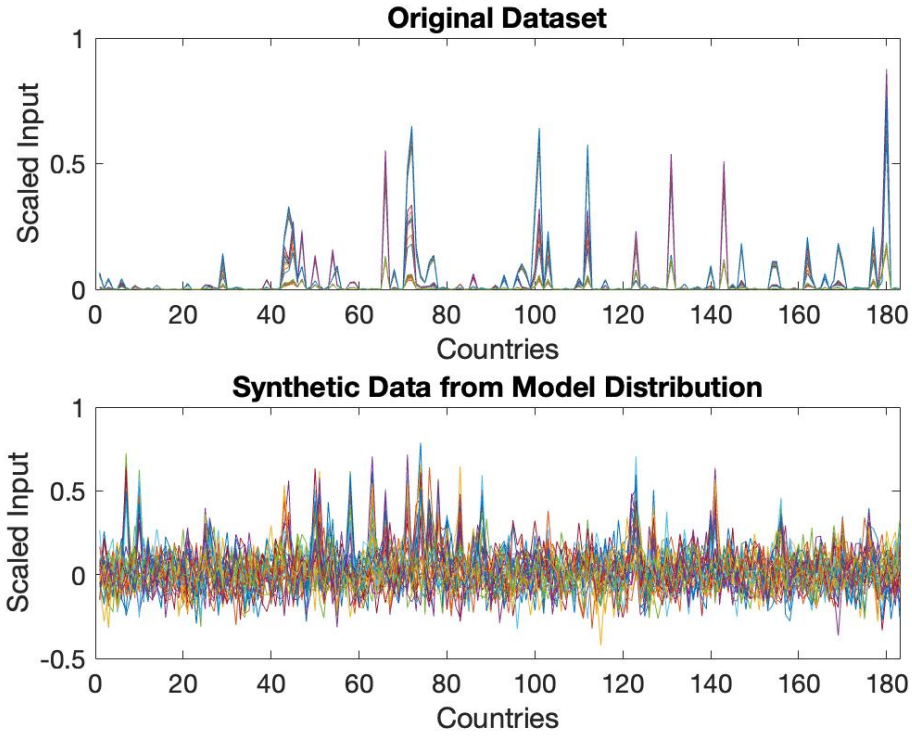

Figure 10.2: Training Dataset

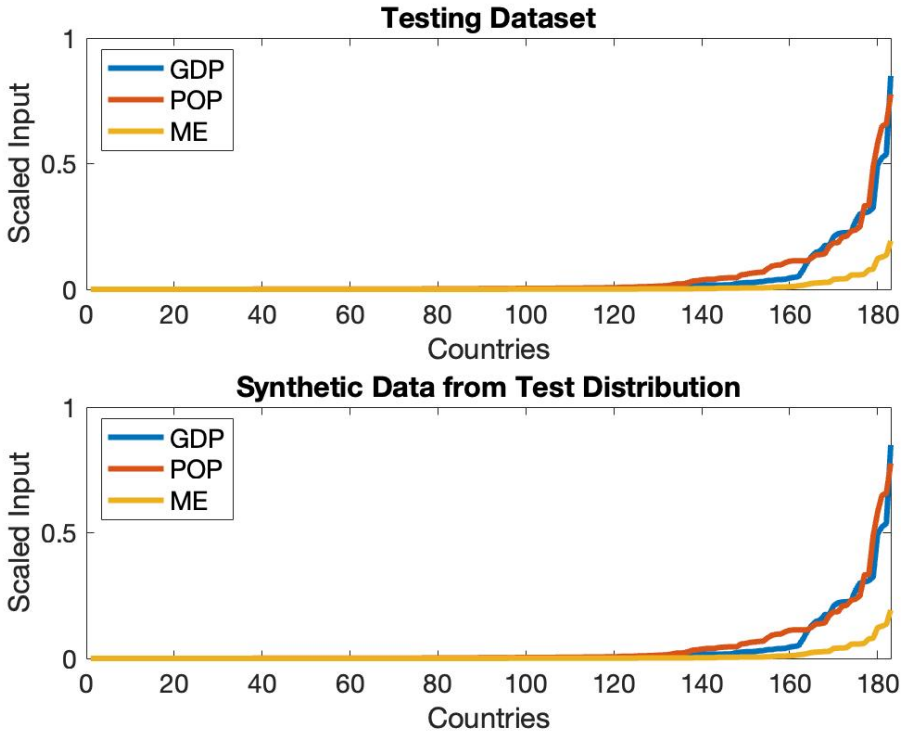

Figure 10.3: Testing Dataset

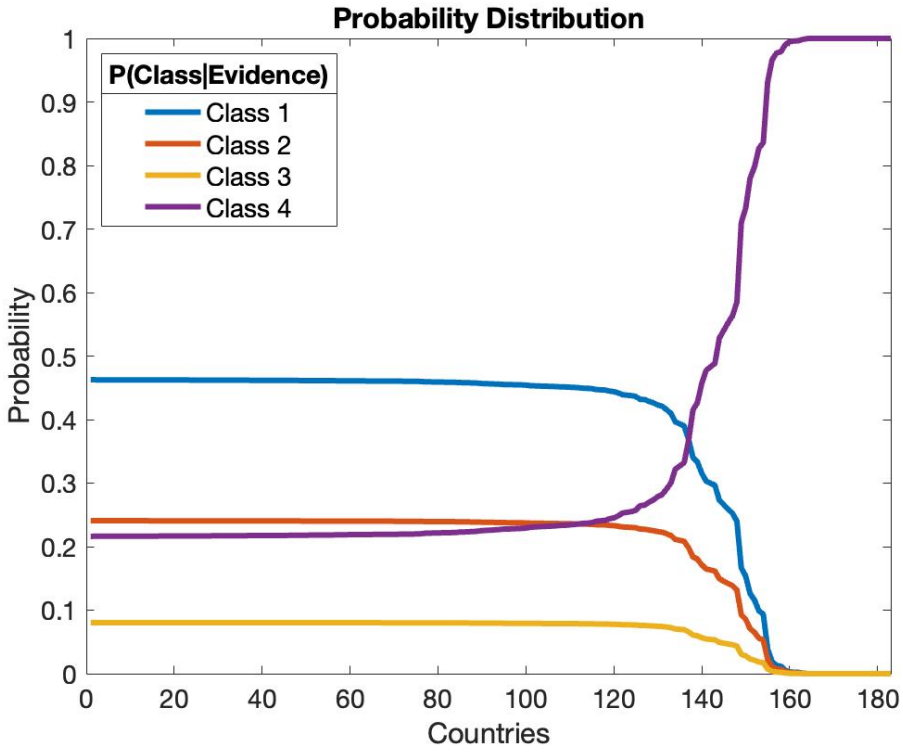

Figure 10.4: Probability Distribution of the Classes

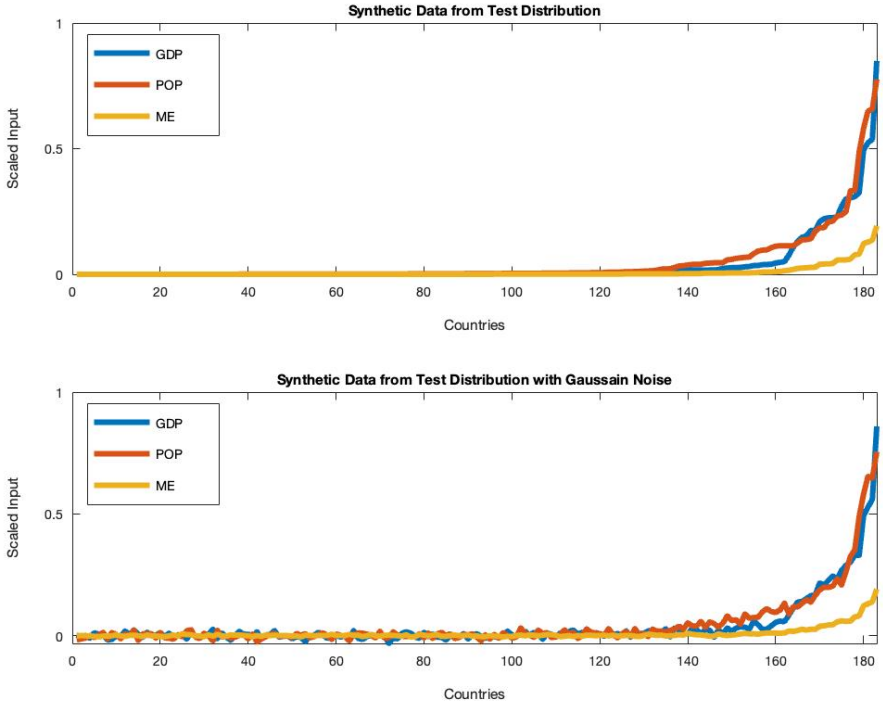

Figure 10.5: Testing Data with and without Gaussian Noise

Chapter 11

Dynamics for Relevant Variables in SEM

The SEM (2.6) gives a stationary correlation relationship between relevant factors in escalation dynamics. However, it does not demonstrate any dynamic effects that these factors may exhibit during the nuclear race escalation process. In this thesis, the focus is on the linearized, first-order approximation of nuclear competing dynamics at the bottom layer. To characterize dynamic evolution of these factors in the first order, first view (2.6) as a limiting equation for certain time-varying process, that is,

$$\limsup_{t\to\infty} f_i(x_i(t)) = 0 = -\limsup_{t\to\infty} x_i(t) \tag{11.1}$$

$$+ \sum_{(i,j)\in E} a_{ij} \limsup_{t\to\infty} x_j(t) + \limsup_{t\to\infty} u_i(t), \quad i = 1, \ldots, n \tag{11.2}$$

for some function $f_i : \mathbb{R} \to \mathbb{R}$, under the assumption that $-\infty < \limsup_{t\to\infty} x_i(t) < +\infty$ and $\limsup_{t\to\infty} u_i(t) = u_i$.

Focusing on first-order dynamics, $f_i(\cdot)$ can be chosen as

$$f_i(x) = \frac{dx}{dt} + A_i x \tag{11.3}$$

where $A_i \geq 0$ is a parameter to be determined later. In this case, dropping off the limit

operation on both sides of (11.2) yields

$$\frac{dx_i(t)}{dt} + A_i x_i(t) = -x_i(t) + \sum_{(i,j)\in E} a_{ij} x_j(t) + u_i(t), \quad t \geq 0, \quad i = 1, \ldots, n \tag{11.4}$$

Rearranging (11.4) yields

$$\frac{dx_i(t)}{dt} = -\Big(A_i + 1 - \sum_{(i,j)\in E} a_{ij}\Big) x_i(t) + \sum_{(i,j)\in E} [a_{ij} x_j(t) - a_{ij} x_i(t)] + u_i(t), \tag{11.5}$$

$$t \geq 0, \quad i = 1, \ldots, n \tag{11.6}$$

Letting $\sigma_{ii} := A_i + 1 - \sum_{(i,j)\in E} a_{ij}$. Then

$$\frac{dx_i(t)}{dt} = -\sigma_{ii} x_i(t) + \sum_{(i,j)\in E} a_{ij}[x_j(t) - x_i(t)] + u_i(t), \quad t \geq 0, \quad i = 1, \ldots, n \tag{11.7}$$

which has the form of compartmental models. Now choose A_i to satisfy

$$A_i \geq \Big|1 - \sum_{(i,j)\in E} a_{ij}\Big|, \quad i = 1, \ldots, n \tag{11.8}$$

so that $\sigma_{ii} \geq 0$. This implies that the rate of change for x_i always has a dissipation term $-\sigma_{ii} x_i$ to counter its growth and to ensure the finiteness of $\limsup_{t\to\infty} x_i(t)$. Moreover, the term $\sum_{(i,j)\in E} a_{ij}[x_j(t) - x_i(t)]$ is the cooperative component, which can be rewritten as a potential-based cooperative game as follows

$$\sum_{(i,j)\in E} a_{ij}[x_j(t) - x_i(t)] = \frac{\partial}{\partial x_i(t)} \left(\frac{1}{2} \sum_{(i,j)\in E} a_{ij}[x_j(t) - x_i(t)]^2 \right) \tag{11.9}$$

The term $-\sigma_{ii} x_i(t)$ has an inhibitory effect [9] in the network, which can be viewed as the compromise component. In this thesis, the model (11.7) is used to depict intertwined dynamics among relevant variables at the bottom level.

Chapter 12

Developing a topological energy level method to draw the energy-like level contour of interested variables for visualization

The proposed multiagent multilayer model has a capability of projecting interested contributors and related variables onto some topological planes for visual illustration of such intertwined dynamics. The first method is to map relevant variables onto some topological planes at the existing model framework. This job can be done through a straightforward way since the proposed model has a linear matrix structure. The second method is a dynamical system and control theory inspired approach by finding multi-level energy cost curves. In this approach, we view the proposed model as a dynamical system governed by differential and/or difference equations. According to the Hamilton's principle in dynamical system theory and Nash equilibria in non-cooperative game theory, the possible propagating dynamics for the proposed model is the one that minimizes the energy cost associated with the Hamiltonian or energy Casimir function. Hence, by plotting the different energy-like levels, one can visualize how the escalation dynamics evolves over time and space.

The matrix decomposition method for visualization of static multiagent network models is as follows. Consider the matrix form of the SEM (2.6). Then

$$x = Wx + u$$

It is known in matrix analysis that the QR decomposition of W can lead to

$$W = QHQ^{\mathrm{T}}$$

where $Q \in \mathbb{R}^{n\times n}$ is an orthogonal matrix and $H \in \mathbb{R}^{n\times n}$ is an upper Hessenberg matrix, i.e., H has the following specific form:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \cdots & \cdots & h_{1n} \\ h_{21} & h_{22} & \ddots & \ddots & \ddots & h_{2n} \\ 0 & h_{32} & \ddots & \ddots & \ddots & h_{3n} \\ 0 & 0 & h_{43} & \ddots & \ddots & h_{4n} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & h_{n,n-1} & h_{nn} \end{bmatrix}$$

Hence,

$$Q(I_n - H)Q^{\mathrm{T}}x = u$$
$$(I_n - H)Q^{\mathrm{T}}x = Q^{\mathrm{T}}u$$

Now let the coordinate transformation be

$$\bar{x} = Q^{\mathrm{T}}x, \quad \bar{u} = Q^{\mathrm{T}}u$$

Then

$$(I_n - H)\bar{x} = \bar{u} \tag{12.1}$$

that is,

$$\begin{bmatrix} 1-h_{11} & -h_{12} & -h_{13} & \cdots & \cdots & -h_{1n} \\ -h_{21} & 1-h_{22} & \ddots & \ddots & \ddots & -h_{2n} \\ 0 & -h_{32} & \ddots & \ddots & \ddots & -h_{3n} \\ 0 & 0 & -h_{43} & \ddots & \ddots & -h_{4n} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & -h_{n,n-1} & 1-h_{nn} \end{bmatrix} \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \vdots \\ \vdots \\ \bar{x}_{n-1} \\ \bar{x}_n \end{bmatrix} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \vdots \\ \vdots \\ \bar{u}_{n-1} \\ \bar{u}_n \end{bmatrix} \tag{12.2}$$

Expanding this equation componentwise yields

$$(1-h_{11})\bar{x}_1 - h_{12}\bar{x}_2 - h_{13}\bar{x}_3 - \cdots - h_{1,n-1}\bar{x}_{n-1} - h_{1n}\bar{x}_n = \bar{u}_1 \tag{12.3}$$

$$-h_{21}\bar{x}_1 + (1-h_{22})\bar{x}_2 - h_{23}\bar{x}_3 - \cdots - h_{2,n-1}\bar{x}_{n-1} - h_{2n}\bar{x}_n = \bar{u}_2 \tag{12.4}$$

$$-h_{32}\bar{x}_2 + (1-h_{33})\bar{x}_3 - \cdots - h_{3,n-1}\bar{x}_{n-1} - h_{3n}\bar{x}_n = \bar{u}_3 \tag{12.5}$$

$$\vdots$$

$$-h_{n-1,n-2}\bar{x}_{n-2} + (1-h_{n-1,n-1})\bar{x}_{n-1} - h_{n-1,n}\bar{x}_n = \bar{u}_{n-1} \tag{12.6}$$

$$-h_{n,n-1}\bar{x}_{n-1} + (1-h_{nn})\bar{x}_n = \bar{u}_n \tag{12.7}$$

Thus, for fixed $\bar{u}_i$, these equations generate multiple hyperplanes that can be used for visualization. This process is recursively going backward based on these equations. More specifically, it starts with the bottom equation

$$-h_{n,n-1}\bar{x}_{n-1} + (1-h_{nn})\bar{x}_n = \bar{u}_n$$

This equation gives a straight line for 2D visualization of $(\bar{x}_{n-1}, \bar{x}_n)$ under fixed $\bar{u}_n$. Moving

up, the next equation is

$$-h_{n-1,n-2}\bar{x}_{n-2} + (1 - h_{n-1,n-1})\bar{x}_{n-1} - h_{n-1,n}\bar{x}_n = \bar{u}_{n-1}$$

which gives a plane for 3D visualization of $(\bar{x}_{n-2}, \bar{x}_{n-1}, \bar{x}_n)$ under fixed $\bar{u}_{n-1}$. Following this procedure with some variable elimination, one can obtain a 2D or 3D visualization for any pair $(\bar{x}_i, \bar{x}_j)$ or triple $(\bar{x}_i, \bar{x}_j, \bar{x}_k)$ in the transformed space, $i \neq j \neq k$, $i, j, k = 1, \ldots, n$. However, this method still does not solve the problem of visualizing (x_i, x_j) or (x_i, x_j, x_k) in the original space, $i \neq j \neq k$, $i, j, k = 1, \ldots, n$.

If $I_n - H$ is nonsingular, then we have

$$x = Q\bar{x} = Q(I_n - H)^{-1}Q^{\mathrm{T}}u$$

which means that for given u, such x is uniquely determined. Let $\mathbf{e}_i \in \mathbb{R}^n$ denotes the column vector whose ith element is 1 and the rest are zero. Then it follows that

$$x_i = \mathbf{e}_i^{\mathrm{T}}x = \mathbf{e}_i^{\mathrm{T}}Q(I_n - H)^{-1}Q^{\mathrm{T}}u$$

Therefore, visualizing (x_i, x_j) or (x_i, x_j, x_k) becomes plotting

$$\left(\mathbf{e}_i^{\mathrm{T}}Q(I_n - H)^{-1}Q^{\mathrm{T}}u, \mathbf{e}_j^{\mathrm{T}}Q(I_n - H)^{-1}Q^{\mathrm{T}}u\right)$$

or

$$\left(\mathbf{e}_i^{\mathrm{T}}Q(I_n - H)^{-1}Q^{\mathrm{T}}u, \mathbf{e}_j^{\mathrm{T}}Q(I_n - H)^{-1}Q^{\mathrm{T}}u, \mathbf{e}_k^{\mathrm{T}}Q(I_n - H)^{-1}Q^{\mathrm{T}}u\right)$$

as $u = \sum_{i=1}^{n} \mathbf{e}_i u_i$ varies in the space span $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$. The invertibility check of $I_n - H$ would be easier due to the upper Hessenberg form of H.

If $I_n - H$ is singular, then we have to use the above derived equations to visualize the

original SEM model. This is due to the fact that the solution to $(I_n - H)\bar{x} = \bar{u}$ in terms of $\bar{x}$ is not unique for given $\bar{u}$. In this case, start with (12.7) by fixing $\bar{u}$ and $\bar{x}_n$ to solve $\bar{x}_{n-1}$. Then moving up to (12.6) to obtain $\bar{x}_{n-2}$. Bottoming up this process will lead to $\bar{x}$ for given $\bar{u}$ and $\bar{x}_n$. Next, incrementally change the value of $\bar{x}_n$, and repeat this process to obtain another $\bar{x}$. After several rounds of computation, a series of $\bar{x}$ denoted by $\bar{x}[l]$, $l = 1, \ldots, N$ is obtained. Then plotting $(\mathbf{e}_i^{\mathrm{T}} Q\bar{x}[l], \mathbf{e}_j^{\mathrm{T}} Q\bar{x}[l])$ or $(\mathbf{e}_i^{\mathrm{T}} Q\bar{x}[l], \mathbf{e}_j^{\mathrm{T}} Q\bar{x}[l], \mathbf{e}_k^{\mathrm{T}} Q\bar{x}[l])$ sequentially for $l = 1, \ldots, N$ will generate the 2D or 3D map for visualization of (x_i, x_j) or (x_i, x_j, x_k).

The energy-level method for visualization of dynamic multiagent network models is as follows. The proposed energy-level method for visualization targets various dynamic models as first-order differential equations in different layers. It uses the concept of energy-like functions such as Casimir or Hamiltonian functions in dynamical systems and optimal control theory, and maps the differential dynamics onto a manifold defined by contours of Casimir or Hamiltonian functions. This can help easily visualize the change of competing dynamics along the gradient direction of Casimir or Hamiltonian functions in the simulation. In this project, focus is on the first-order dynamics (11.7). The key here is to find such an appropriate Casimir or Hamiltonian function. To this end, a new notion of energy-Casimir functions to facilitate such a searching process.

Definition 12.0.1. *A continuously differentiable function $C : \mathbb{R}^n \to \mathbb{R}$ is an energy-Casimir function of (11.7) if it is nonincreasing along the flow of (11.7) for some fixed $u \equiv u_0 \in \mathbb{R}^n$, that is,*

$$C'(x) f(x, u)\Big|_{u \equiv u_0} \leq 0$$

where $C'(x) = \frac{\partial C(x)}{\partial x}$, $x = [x_1, \ldots, x_n]^{\mathrm{T}} \in \mathbb{R}^n$ denotes the state vector for (11.7), $u = [u_1, \ldots, u_n]^{\mathrm{T}} \in \mathbb{R}^n$ denotes the input vector for (11.7), $f(x, u) = [f_1(x, u_1), \ldots, f_n(x, u_n)]^{\mathrm{T}}$,

and

$$f_i(x, u_i) = -\sigma_{ii} x_i + \sum_{(i,j)\in E} a_{ij}(x_j - x_i) + u_i$$

The following result gives a way to find appropriate energy-Casimir functions.

Theorem 12.0.1. *If there exist an $n \times n$ symmetric matrix $P = P^{\mathrm{T}}$ and an $n \times n$ positive semidefinite matrix $Q = Q^{\mathrm{T}} \geq 0$ such that for every $x \in \mathbb{R}^n$,*

$$\frac{\partial f(x, u_0)}{\partial x} P + P \left(\frac{\partial f(x, u_0)}{\partial x} \right)^{\mathrm{T}} \leq -Q \tag{12.8}$$

holds for some fixed $u_0 \in \mathbb{R}^n$, then $C(x) = f^{\mathrm{T}}(x, u_0) P f(x, u_0)$ is an energy-Casimir function.

Proof: Note that the time derivative of $C(x) = f^{\mathrm{T}}(x, u_0) P f(x, u_0)$ along the trajectories of (11.7) with $u \equiv u_0$ is given by

$$\begin{aligned}
\dot{C}(x) &= C'(x) f(x, u_0) \\
&= 2 f^{\mathrm{T}}(x, u_0) P \frac{\partial f(x, u_0)}{\partial x} f(x, u_0) \\
&= f^{\mathrm{T}}(x, u_0) \left[\frac{\partial f(x, u_0)}{\partial x} P + P \left(\frac{\partial f(x, u_0)}{\partial x} \right)^{\mathrm{T}} \right] f(x, u_0) \\
&\leq -f^{\mathrm{T}}(x, u_0) Q f(x, u_0) \\
&\leq 0
\end{aligned}$$

Hence, by definition, $C(x) = f^{\mathrm{T}}(x, u_0) P f(x, u_0)$ is an energy-Casimir function for (11.7). ■

Note that $C(x) = f^{\mathrm{T}}(x, u_0) P f(x, u_0)$ can be written as $C(x) = \dot{x}^{\mathrm{T}} P \dot{x}$ for (11.7) with $u \equiv u_0$. Since P is symmetric, it has the diagonal decomposition form $P = S^{\mathrm{T}} \mathrm{diag}(\lambda_1, \ldots, \lambda_n) S$. Hence, $C(x) = \sum_{i=1}^{n} \lambda_i \dot{\kappa}_i^2$, where $[\dot{\kappa}_1, \ldots, \dot{\kappa}_n]^{\mathrm{T}} = S\dot{x}$. Therefore, the physical meaning of this energy-Casimir function is the "kinetic energy" of the system (11.7).

Since $f(x, u_0)$ can be written as

$$f(x, u_0) = -Ax - x + Wx + u_0$$

where $A = \text{diag}(A_1, \ldots, A_n)$. Then

$$\frac{\partial f(x, u_0)}{\partial x} = -A - I_n + W^{\mathrm{T}}$$

Hence, (12.8) becomes

$$(-A - I_n + W^{\mathrm{T}})P + P(-A - I_n + W) + Q \leq 0 \tag{12.9}$$

which is a linear matrix inequality (LMI). This inequality can be solved by using the MATLAB LMI toolbox. The following result gives a sufficient condition for this LMI to have a unique solution.

Theorem 12.0.2. *Let $C \in \mathbb{R}^{l \times n}$. If the pair $(-A - I_n + W, C)$ is observable, then*

$$(-A - I_n + W^{\mathrm{T}})P + P(-A - I_n + W) + Q = 0 \tag{12.10}$$

has a unique solution $P = P^{\mathrm{T}} > 0$, where $Q = C^{\mathrm{T}}C$.

Proof: This is a standard result for the Lyapunov equation in control theory. Hence, the proof is omitted. ■

The observability of $(-A - I_n + W, C)$ can be checked via the rank condition or the PBH test in control theory as follows.

Theorem 12.0.3. *The following statements are equivalent:*

1) The pair $(-A - I_n + W, C)$ is observable.

2) For any $\lambda \in \mathbb{C}$, the rank of $\begin{bmatrix} sI_n + A + I_n - W \\ C \end{bmatrix}$ *is n.*

3) the rank of $\begin{bmatrix} C \\ C(-A - I_n + W) \\ \vdots \\ C(-A - I_n + W)^{n-1} \end{bmatrix}$ *is* n.

Proof: This is a standard result in control theory. Hence, the proof is omitted. ■

Since there is freedom to choose A, it is possible to satisfy the observability condition for given C. For example, $C = I_n$ always satisfies the observability of $(-A - I_n + W, I_n)$.

The energy-Casimir method describes the dynamic evolution of "kinetic energy" of the system. However, it does not include any indication on the system state performance. Next, another energy-like function is presented, Hamiltonian function. As an enhanced version of energy-Casimir functions to embed the information of system state performance. The idea of this method comes from optimal control theory where a Hamiltonian function is used to find out optimal control laws via the Hamilton-Jacobi-Bellman equation.

To find a Hamiltonian function, the first step is to define a performance cost function for (11.7), which is given by an integral form

$$J = \int_{t_0}^{\infty} L(x(t), u(t))dt \tag{12.11}$$

where t_0 is the initial time, $x(t) \in \mathbb{R}^n$ denotes the state vector for (11.7), and $u(t) \in \mathbb{R}^n$ denotes the input vector for (11.7). The choice of $L(x, u)$ could be quadratic, e.g., $L(x, u) = x^{\mathrm{T}} R_1 x + u^{\mathrm{T}} R_2 u$ for $R_1 = R_1^{\mathrm{T}} \geq 0$ and $R_2 = R_2^{\mathrm{T}} \geq 0$, or non-quadratic, e.g., $L(x, u) = P_1 x + P_2 u$, or the mixture of the quadratic and non-quadratic terms.

Definition 12.0.2. *Let* $p \in \mathbb{R}^n$. *Then* $H(x, u, p) = L(x, u) + p^{\mathrm{T}} f(x, u)$ *is called a Hamiltonian function for (11.7).*

This definition is quite general since p is arbitrary. To narrow down the search for a useful Hamiltonian function, let $C : \mathbb{R}^n \to \mathbb{R}$ be an energy-Casimir function. Then $H(x, u) = L(x, u) + C'(x) f(x, u)$ is an energy-Hamiltonian function for (11.7) associated

with the energy-Casimir function $C(x)$. It embeds both $C(x)$ and $L(x,u)$ into a single expression for visualization.

Let $H(x) := H(x,u)|_{u\equiv u_0}$. Then the pair (x_i, x_j) can be visualized in 2D space or the triple (x_i, x_j, x_k) in 3D space by plotting the energy-Casimir level set

$$\left\{(x_i,x_j)\in\mathbb{R}^2 : C(x_1,\ldots,x_i,\ldots,x_j,\ldots,x_n)\Big|_{x_l=c_l\in\mathbb{R},l\neq i,j,l=1,\ldots,n} = c\in\mathbb{R}\right\} \tag{12.12}$$

$$\left\{(x_i,x_j,x_k)\in\mathbb{R}^3 : C(x_1,\ldots,x_i,\ldots,x_j,\ldots,x_k,\ldots,x_n)\Big|_{x_l=c_l\in\mathbb{R},l\neq i,j,k,l=1,\ldots,n} = c\in\mathbb{R}\right\} \tag{12.13}$$

or the energy-Hamiltonian level set

$$\left\{(x_i,x_j)\in\mathbb{R}^2 : H(x_1,\ldots,x_i,\ldots,x_j,\ldots,x_n)\Big|_{x_l=c_l\in\mathbb{R},l\neq i,j,l=1,\ldots,n} = c\in\mathbb{R}\right\} \tag{12.14}$$

$$\left\{(x_i,x_j,x_k)\in\mathbb{R}^3 : H(x_1,\ldots,x_i,\ldots,x_j,\ldots,x_k,\ldots,x_n)\Big|_{x_l=c_l\in\mathbb{R},l\neq i,j,k,l=1,\ldots,n} = c\in\mathbb{R}\right\} \tag{12.15}$$

Chapter 13

Conclusion

This thesis outlined a way to describe the spatial-temporal evolution and visualization of escalation dynamics with multiple players. In this project, multiagent coordination and algebraic graph theory based on the advisor's prior research [14] was implemented to set up an interconnected, graphical representation of multiple agents interacting within a hybrid game. The hybrid game, which consists of non-cooperative gaming as the primary dynamic, and cooperative gaming as the secondary dynamic, was used to capture the multi-modal nature of escalation dynamics. This led to a multilayer interconnected complex system to model intrinsic dynamics of competing escalation.

In addition to the hybrid game, a three layer approach to prepare data before being input into the model was constructed. The three layers included a retrieving network, an analyzing network, and a formulating network. The hybrid game model associated with these factors resulting from the three layer network provided alternative interdependencies among different agents in the hybrid game: cooperative (pro) interaction and competing (con) interaction. This hybrid game provided the perspective of the two sides of impact from their input: cooperative component and non-cooperative component. The cooperative component, reflected the steady, coordinated nuclear deployment strategy in the escalation and was the result of the potential-based cooperative game. The non-cooperative component, depicted the negative contribution due to adversarial effects and competitive dynamics in escalation, was the result of inhibitory effects in the network model. Finally, the overall

outcome of the hybrid game was the results from both games weighted by their priority. The weight of this priority is determined by the multi-cue multi-choice (MCMC) decision making [10]. This allowed the network to govern the process of network model adaptation by mimicking how a human makes decisions in these same scenarios. Finally, a topological energy level method was developed to draw the energy-like level contour of the network model for visualization by the user of the system.

Simulations provided in this thesis were used to verify the mathematical models were implemented correctly and performed as expected. The hybrid game was simulated to verify the cooperative and non-cooperative aspects of each game gave convergent and expected results. The MCMC model and analzing network were also simulated to verify their effectiveness. At the time of writing the retrieving network, formulating network, and visualizations were not yet simulated. However, from the simulations completed it was verified that the hybrid gaming, MCMC model, and analyzing networks performed as expected. Upon the completion of the remaining simulations, a software package that encompasses each of these aspects will be formulated. The software package can be used to describe the escalation dynamics among different nation in escalation scenarios.

Bibliography

[1] J. Baffes, M. C. Hyland, and B. Blankespoor. World bank open data, Dec 2022.

[2] R. Bogacz, E. Brown, J. Moehlis, P. Holmes, and J. D. Cohen. The physics of optimal decision making: A formal analysis of models of performance in two-alternative forced-choice tasks. *Psychological Review*, 113(4):700, 2006.

[3] E. Bonabeau. Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(3):7280–7287, 2002.

[4] A. Darwiche. A differential approach to inference in Bayesian networks. In *Uncertainty in Artificial Intelligence Proceedings 2000*, pages 123–132, 2000.

[5] M. Davis and M. Maschler. The kernel of a cooperative game. *Naval Research Logistics*, 12(3):223–259, 1965.

[6] G. Deco, E. T. Rolls, L. Albantakis, and R. Romo. Brain mechanisms for perceptual and reward-related decision-making. *Progress in Neurobiology*, 103:194–213, 2013.

[7] V. Draglia, A. G. Tartakovsky, and V. V. Veeravalli. Multihypothesis sequential probability ratio tests–Part I: Asymptotic optimality. *IEEE Transactions on Information Theory*, 45(7):2448–2461, 1999.

[8] V. Draglia, A. G. Tartakovsky, and V. V. Veeravalli. Multihypothesis sequential probability ratio tests–Part II: Accurate asymptotic expansions for the expected sample size. *IEEE Transactions on Information Theory*, 46(4):1366–1383, 2000.

[9] M. Firouznia. *Human-Cognition-in-the-Loop: A Framework to Include Decision Process in Closed Loop Control*. PhD thesis, Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, Lincoln, NE, July 2019.

[10] M. Firouznia and Q. Hui. On performance gauge of average multi-cue multi-choice decision making: A converse Lyapunov approach. *IEEE/CAA Journal of Automatica Sinica*, 8(1):136–147, 2021.

[11] M. Firouznia, C. Peng, and Q. Hui. Toward building a human-cognition-in-the-loop supervisory control system for humanized decision-making. In *The 13th Annual IEEE International Systems Conference*, Orlando, FL, April 2019.

[12] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, New York, 2001.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proceedings of the International Conference on Neural Information Processing Systems*, pages 2672–2680, 2014.

[14] Q. Hui and H. Zhang. Optimal balanced coordinated network resource allocation using swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(5):770–787, 2015.

[15] S. Ioannidis and P. Loiseau. Linear regression as a non-cooperative game. In *Proceedings of the 9th International Conference on Web and Internet Economics*, pages 277–290, 2013.

[16] J. R. Marden, G. Arslan, and J. S. Shamma. Cooperative control and potential games. *IEEE Transactions on Systems, Man, and Cybernetics–Part B: Cybernetics*, 39(6):1393–1407, 2009.

[17] T. McMillen and P. Holmes. The dynamics of choice among multiple alternatives. *Journal of Mathematical Psychology*, 50(1):30–57, 2006.

[18] T. M. Mitchell. *Machine learning.* McGraw-Hill, 1997.

[19] J. Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2):286–295, 1951.

[20] A. Roxin. Drift-diffusion models for multiple-alternative forced-choice decision making. *Journal of Mathematical Neuroscience*, 9(5), 2019.

[21] M. Usher and J. L. McClelland. The time course of perceptual choice: The leaky, competing accumulator model. *Psychological Review*, 108(3):550, 2001.

[22] H. Zhang. *Coordinated resource allocation and load balancing for network systems using semistability tools and multiagent coordination.* PhD thesis, Department of Mechanical Engineering, Texas Tech University, Lubbock, TX, 2014.

[23] H. Zhang, Q. Hui, and Q. Liu. Bio-inspired consensus under suggested convergence direction. In *2017 Amer. Control Conf.*, Seattle, WA, 2017.