ADAPTIVE GENETIC ALGORITHMS FOR PARTICLE FILTERING IMPROVEMENT

CHANIN KUPTAMETEE

DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

SCHOOL OF APPLIED DIGITAL TECHNOLOGY
MAE FAH LUANG UNIVERSITY
2025

ADAPTIVE GENETIC ALGORITHMS FOR PARTICLE FILTERING IMPROVEMENT

CHANIN KUPTAMETEE

THIS DISSERTATION IS A PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN COMPUTER ENGINEERING

SCHOOL OF APPLIED DIGITAL TECHNOLOGY
MAE FAH LUANG UNIVERSITY
2025

DISSERTATION APPROVAL

MAE FAH LUANG UNIVERSITY

FOR

DOCTOR OF PHILOSOPHY IN COMPUTER ENGINEERING

Dissertation Title: Adaptive Genetic Algorithms for Particle Filtering Improvement

Author: Chanin Kuptametee

Examination Committee:

Professor Prayoot Akkaraekthalin, Ph. D.	Chairperson
Associate Professor Nattapol Aunsri, Ph. D.	Member
Professor Zoi-Heleni Michalopoulou, Ph. D.	Member
Associate Professor Punnarumol Temdee, Ph. D.	Member
Associate Professor Roungsan Chaisricharoen, Ph. D.	Member
Assistant Professor Chayapol Kamyod, Ph. D.	Member

Advisors:

.....Advisor

(Associate Professor Nattapol Aunsri, Ph. D.)

.....Co-Advisor

(Professor Zoi-Heleni Michalopoulou, Ph. D.)

Dean:

.....

(Associate Professor Nacha Chondamrongkul, Ph. D.)

ACKNOWLEDGEMENTS

First of all, I would like to deeply appreciate my dissertation advisor, Assoc. Prof. Nattapol Aunsri, Ph. D., for his expertise, encouragement, support, and patience, devoted to this dissertation. I am also deeply grateful to my co-advisor, Prof. Zoi-Heleni Michalopoulou, Ph. D., for her patience devoted to guidance and writing corrections throughout the research.

I would like to express gratitude to all examiners, Prof. Prayoot Akkaraekthalin, Ph. D., Assoc. Prof. Punnarumol Temdee, Ph. D., Assoc. Prof. Roungsan Chaisricharoen, Ph. D., and Asst. Prof. Chayapol Kamyod, Ph. D., for valuable feedback and suggestions for reshaping this dissertation.

I also would like to be grateful to all instructors that I took their classes during my Ph. D. journey at School of Applied Digital Technology (formerly School of Information Technology), Mae Fah Luang University, Asst. Prof. Gp. Capt. Thongchai Yooyativong, Ph. D., Assoc. Prof. Punnarumol Temdee, Ph. D., Assoc. Prof. Roungsan Chaisricharoen, Ph. D., Assoc. Prof. Nattapol Aunsri, Ph. D., and Asst. Prof. Chayapol Kamyod, Ph. D., for their immense knowledge and dedication in teaching.

I would like to acknowledge Mae Fah Luang University for the graduate scholarship (Grant No. 008) and the dissertation support grant (Grant No. 0245). I also would like to be grateful to all staffs of Office of the Postgraduate Studies and School of Applied Digital Technology for their assistance and support. Last but not least, I would like to express gratitude to my beloved family for continuous love and support at any time.

Chanin Kuptametee

Dissertation Title	Adaptive Genetic Algorithms for Particle Filtering Improvement
Author	Chanin Kuptametee
Degree	Doctor of Philosophy (Computer Engineering)
Advisor	Associate Professor Nattapol Aunsri, Ph. D.
Co-Advisor	Professor Zoi-Heleni Michalopoulou, Ph. D.

ABSTRACT

Particle filtering is a scheme under sequential Bayesian framework widely employed to estimate state of desired information from the observation data outputted from non-linear, non-Gaussian systems. We proposed an adaptive genetic algorithm-based scheme to enhance quality of the drawn sample vectors of state variables (called particles). Each low-weight parent pairs with a randomly selected high-weight parent. The newly created offspring particle is allowed to replace its low-weight parent only if the weight of the offspring is higher than the weight of the low-weight parent. The accepted offspring particles with high weights can also be paired with the other low-weight parents in order to promote particle diversity. Simulation results show that the new method is superior to state-of-the-art algorithms in estimating one-dimensional and multidimensional state estimation. The new method is also tested in an application under the multiple-model particle filter (MMPF) framework of spectrum and dispersion curve estimation of a time-varying acoustics propagated through an ocean waveguide. The new method still can perform well in capturing the modal frequency. However, the new method is also sensitive to high-intensity time-domain noise where such severe noise causes false frequency contents to be more likely to be misidentified as modal frequency. Such a pilot study of testing the new method on the MMPF indicates that further research and improvements of GAs still be needed.

Keywords: Genetic Algorithm, Particle Degeneracy, Particle Diversity, Particle Filter, Particle Impoverishment, Resampling, Time-frequency Representation

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES

LIST OF FIGURES

ABBREVIATIONS AND SYMBOLS

1-D	One-dimensional
CDF	Cumulative Density Function
dB	Decibel
df	Degrees of Freedom
$\delta(\cdot)$	Dirac Delta Function
e.g.	Exempli Gratia
GA	Genetic Algorithm
Hz	Hertz
i.e.	Id Est
i.i.d.	Independent and Identically Distributed
KF	Kalman Filter
MAE	Mean Absolute Error
MAP	Maximum a Posteriori
MMPF	Multiple-Model Particle Filter
ms	Millisecond
PDF	Probability Density Function
PF	Particle Filter
PMF	Probability Mass Function
RMSE	Root-mean-squared Error
SIR	Sequential Importance Resampling
SIS	Sequential Importance Sampling
SMC	Sequential Monte Carlo
SNR	Signal-to-noise Ratio
STFT	Short-time Fourier Transform
TFR	Time-frequency Representation
WM	Weighted Mean

CHAPTER 1

INTRODUCTION

1.1 Research Rationale

In scientific and engineering problems and applications, we need to obtain the state of the desired information (e.g., target parameters) from any system. These parameters, however, cannot be measured directly because they are hidden in observation data. Also, the hidden state can be assumed time-varying because the observation values can change as time goes. To extract the state of targeted parameters from the sequential observation data, we need a function that relates the state parameters and the observation data together. While such a function can be designed according to our related prior knowledge, there can be uncertainties (or random noise) that occur during the output measurements. A direct inversion process that finds the state by inverting the designed function and employing the observation data can be inefficient (Cappé et al., 2007; Krumm, 2010). Therefore, estimating states of time-varying parameters from noisy sequential observations is a challenging but important task in order to understand the nature of any system.

Bayesian approaches find the probability density function (PDF) of targeted parameters that is conditional on observation data. This PDF is called the posterior PDF because we must first obtain the observation data before the PDF can be created (Candy, 2016). However, we cannot obtain the full posterior PDF because it requires all possible state values of the parameters. Also, the posterior PDF does not stay fixed because parameter states evolve with time.

Kalman (1960) proposed the Kalman filtering (KF) as a sequential Bayesian filtering approach that is optimal for systems with a linear relationship between states and the observation data that are corrupted with additive, uncorrelated, and zero-mean normally distributed noise (or additive white zero-mean Gaussian noise). It fully characterizes the Gaussian posterior PDFs by estimating their means and covariances at each time step. There are also variants of KFs proposed to improve the performance

of original KFs. Ensemble KFs (EnKFs) act as the approximation version of the original KF by drawing samples (or ensembles) to estimate the mean and the covariance of the Gaussian posterior PDF at each time step (Evensen, 1994); Katzfuss et al., 2016; van Leeuwan, 2020), while Unscented KFs (UKFs) employ only a few selected samples (or points) to capture the mean and the covariance (Julier, 1997; Wan & van der Merwe, 2000). Extended KFs (EKFs) were proposed to work with non-linear systems by using a linearization processes (Maybeck, 1982). However, KFs are not the optimal methods for estimating states from highly non-linear systems with non-Gaussian posterior PDFs (Candy, 2016; Gordon et al., 1993; Ristic et al., 2004; Roonizi, 2022).

Particle filtering (PF) is a sequential Monte Carlo (SMC) method that randomly draws independent and identically distributed (i.i.d.) sample vectors of values of state variables from the prior PDF of initial state values; these sample vectors are called "particles". Next, we find the importance weight of each particle and normalize these weights to obtain probability masses (or normalized weighted particles), which partially represent the posterior PDF of state given noisy observations. Finally, we infer (or estimate) the hidden state from this approximated posterior PDF (Candy, 2016; Ristic et al., 2004). PF is proved effective in many applications, for example, signal processing (Andrieu et al., 2003; Aunsri & Chamnongthai, 2019; Aunsri & Chamnongthai 2021; Aunsri & Michalopoulou, 2014; Michalopoulou & Aunsri, 2018; Yardim et al., 2011; Zorych & Michalopoulou, 2008), agriculture (Saenmuang & Aunsri, 2019), fault detection (Yin & Zhu, 2015; Yu et al., 2019), moving object tracking (Bhat et al., 2021; Han et al., 2011; Park et al., 2009; Wang et al., 2020), and non-destructive evaluation (Zafar et al., 2020).

Drawing particles in great numbers causes the approximated posterior PDF to get closer to the true PDF, but a higher cost is required. Also, because all particles are randomly drawn, sometimes there can be only a few high-weight particles while the rest have low weights. Consequently, the posterior PDF and state may be poorly estimated (Candy, 2016; Gordon et al., 1993; Ristic et al., 2004). We should reshape the approximated posterior PDF by altering the state values of the particles that are located in low-probability regions in order to relocate them. After we relocate the low-weight particles, their weights must be re-evaluated according to their new state values in order to verify whether or not they become high-weight particles. However, we

should employ the state values of high-weight particles that are available on-hand as clues for finding high-probability regions, instead of relocating the low-weight particles blindly.

Holland (1992) proposed the Genetic algorithm (GA) to imitate the "survival-of-the-fittest" scheme that treats each sample (i.e., state vector) as an individual and performs a selection process to keep high-fit individuals. The high-fit individuals that survive are then employed as parents to produce new offspring state vectors with high diversity among state values (Katoch et al., 2021; Larose, 2006; Michalewicz, 1996). During the offspring creation process, GA blindly creates pairs of two survived particles as parents. In each pair, the state values of two parent vectors are employed to calculate state values of the two new offspring vectors that then replace their parents.

Yin and Zhu (2015) suggested that particles should first be classified as high-weight and low-weight parents. Each low-weight parent must pair with a randomly selected high-weight parent. Only one offspring particle is found from each pair and this offspring particle then replaces its low-weight parent. This ensures existence of high weight parents. However, if the number of high-weight parents is small, diversity of state values of to-be-created offspring particles can be low. Consequently, the chances of discovering new high-probability state values are limited. Also, the weight of the offspring particle may be lower than the weights of their parents. The new set of particles that we obtain after the GA approach is employed may consist of inferior particles whose weights are lower than those of the particles in the old set (or parents' generation). Consequently, the state estimation performance may be unsatisfactory (Kuptametee et al., 2024). Thus, an efficient scheme must be employed in order to ensure that the GA method actually improves the state estimation performance when being integrated in PF algorithms.

1.2 Objectives

1.2.1 To propose a more efficient scheme of employing GA, ensuring quality and diversity of created offspring state vectors in a particle filtering framework.

1.2.2 To employ the developed adaptive GA in a PF algorithm to achieve better state estimation performance for non-linear system, both in one-dimensional and multidimensional systems.

1.3 Scope

1.3.1 This research employs only arithmetic GA operators to improve the performance of the PF framework.

1.3.2 The algorithms and experiments will be implemented in MATLAB.

1.4 Contributions

In our proposed adaptive GA, all original particles (i.e., high-weight parents and low-weight parents) are always prevented from being replaced by inferior offspring particles. This ensures that the state estimation performance will not be degraded. Some offspring particles have weights that can be considered as high according to the threshold employed to classify the original parents before offspring creation. To fix a shortage of high-weight parents, such high-quality offspring particles can then be employed as new high-weight parents. That is, low-weight parents will have more choices of high-weight parents to randomly pair with. Adding new high-weight parents does not increase the complexity of employing the proposed method.

In addition, the proposed adaptive GA does not require too many parameters. Our method can then be employed with ease to enhance performance in any state-space system.

1.5 Dissertation Structure

The remainder of this dissertation is organized as follows:

Chapter 2: Literature Review. This chapter provides the related background theories including sequential Bayesian filtering, PF, and GA. Related previous research is also discussed.

Chapter 3: Proposed Method. This chapter proposes a scheme that ensures efficient employment of an adaptive GA in a PF algorithm.

Chapter 4: Simulation Results. This chapter presents the results of employing the proposed adaptive GA in simulation state-space models. The performance of our proposed method will be compared with that of other state-of-the-art algorithms in cases of a one-dimensional (1-D) and a multidimensional system.

Chapter 5: Application. This chapter presents the results of employing the proposed adaptive GA algorithm to estimate time-varying spectra of a broadband acoustics signal that propagates through the ocean; the likelihood function is non-Gaussian. The experiment is based on the scenario where: (1) the number of frequency modes (or dispersion curves) can vary with time, and (2) the intensity of the additive white Gaussian observation noise that corrupts the time-domain acoustics is unknown.

Chapter 6: Conclusions. This chapter provides conclusions from the overall work. Limitations and future work are also discussed.

CHAPTER 2

LITERATURE REVIEW

2.1 Sequential Bayesian Filtering

In order to estimate hidden states of time-varying targeted parameters of any system, we must first obtain a sequence of observation data as shown in Figure 2.1. In practice, any observation can be corrupted by many kinds of undesired random noise, while observation data can also be time-varying. State-space models are then employed to describe the systems (Candy, 2016). At time step $k \in \mathbf{I}$, let $\mathbf{x}_k \in \mathbf{R}^{d_x}$ be the d_x-dimensional vector of state variables that are hidden in the d_y-dimensional vector of measurable values (i.e., observation) $\mathbf{y}_k \in \mathbf{R}^{d_y}$. There are two functions in the state-space model: (1) the state evolution function $\mathbf{f}_k(\cdot)$ and (2) the observation function $\mathbf{g}_k(\cdot)$. These two functions are not necessarily linear and are respectively expressed as:

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}) \tag{2.1}$$

$$\mathbf{y}_k = \mathbf{g}_k(\mathbf{x}_k, \mathbf{v}_k), \tag{2.2}$$

where $\mathbf{u}_{k-1} \in \mathbf{R}^{d_u}$ is a d_u-dimensional vector of state evolution noise (as independent and identically distributed (i.i.d.) random values) that updates values of state $\mathbf{x}_{k-1}$ to obtain new values $\mathbf{x}_k$, while vector $\mathbf{v}_k \in \mathbf{R}^{d_v}$ is a d_v-dimensional vector of i.i.d. random noise that corrupts the observation $\mathbf{y}_k$ (Candy, 2016; Ristic et al., 2004).

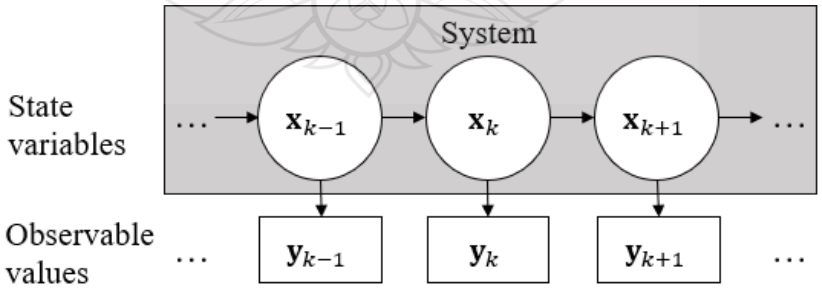


Figure 2.1 State variables hidden in sequential observable data

As previously mentioned, the true state variables cannot be obtained directly due to contaminating observation noise which sometimes can be too severe to be handled with denoising tools. A Bayesian approach is a method employed to find the posterior probability density function (PDF) of state variables conditional on the observation (Candy, 2016). Suppose that we need to find the full posterior PDF at time step k. Let $\mathbf{X}_k = \{\mathbf{x}_1, \dots, \mathbf{x}_k\}$ be a set of all states up to time step k and $\mathbf{Y}_k = \{\mathbf{y}_1, \dots, \mathbf{y}_k\}$ be a set of data obtained from all observations up to time step k. The posterior PDF can be expressed by decomposition via Bayes' rule as:

$$p(\mathbf{X}_k|\mathbf{Y}_k) = \frac{p(\mathbf{X}_k,\mathbf{Y}_k)}{p(\mathbf{Y}_k)} = \frac{p(\mathbf{X}_k,\mathbf{y}_k,\mathbf{Y}_{k-1})}{p(\mathbf{y}_k,\mathbf{Y}_{k-1})} = \frac{p(\mathbf{y}_k|\mathbf{X}_k,\mathbf{Y}_{k-1})p(\mathbf{X}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})}$$

$$p(\mathbf{X}_k|\mathbf{Y}_k) = \frac{p(\mathbf{y}_k|\mathbf{X}_k)p(\mathbf{X}_k|\mathbf{Y}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})}, \tag{2.3}$$

where $p(\mathbf{y}_k|\mathbf{X}_k)$ is the likelihood function expressing the PDF of the observation $\mathbf{y}_k$ conditional on the set of all states $\mathbf{X}_k$ (Candy, 2016). Function $p(\mathbf{X}_k|\mathbf{Y}_{k-1})$ is the prior PDF employed in state prediction and expressed via the Chapman-Kolmogorov equation as:

$$p(\mathbf{X}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_{k-1})p(\mathbf{X}_{k-1}|\mathbf{Y}_{k-1})d\mathbf{X}_{k-1}, \tag{2.4}$$

and $p(\mathbf{y}_k|\mathbf{Y}_{k-1})$ is the normalizing denominator expressed as:

$$p(\mathbf{y}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{y}_k|\mathbf{X}_k, \mathbf{Y}_{k-1})p(\mathbf{X}_k|\mathbf{Y}_{k-1})d\mathbf{X}_k. \tag{2.5}$$

We can assume that all observations in $\mathbf{Y}_k$ are mutually independent from each other because each observation $\mathbf{y}_k$ is contaminated with i.i.d. random noise. Also, each observation $\mathbf{y}_k$ is assumed to be conditional on only the hidden state vector $\mathbf{x}_k$ at the same time step (Candy, 2016). We can then reduce Equation 2.3 as follows:

$$p(\mathbf{X}_k|\mathbf{Y}_k) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{X}_k|\mathbf{Y}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})}. \tag{2.6}$$

After the posterior PDF is obtained, the state can be estimated (or inferred) as the maximum a posteriori (MAP) estimate or the conditional mean (CM) of the posterior PDF expressed as:

$$\widehat{\mathbf{X}}_k^{MAP} = \underset{\mathbf{X}_k}{\operatorname{argmax}}\, p(\mathbf{X}_k|\mathbf{Y}_k) \tag{2.7}$$

and

$$\widehat{\mathbf{X}}_k^{CM} = \int \mathbf{X}_k p(\mathbf{X}_k|\mathbf{Y}_k) d\mathbf{X}_k. \tag{2.8}$$

Alternatively, we can express the full posterior PDF in a recursive form by starting from expressing it via Bayes' rule as:

$$p(\mathbf{X}_k|\mathbf{Y}_k) = \frac{p(\mathbf{Y}_k|\mathbf{X}_k)p(\mathbf{X}_k)}{p(\mathbf{Y}_k)}, \tag{2.9}$$

where the functions $p(\mathbf{Y}_k|\mathbf{X}_k)$, $p(\mathbf{X}_k)$, and $p(\mathbf{Y}_k)$, denote the full likelihood function, the full prior distribution, and the evidence or the normalizing denominator, respectively (Candy, 2016). The full likelihood function $p(\mathbf{Y}_k|\mathbf{X}_k)$ can be decomposed via Bayes' rule as:

$$\begin{aligned} p(\mathbf{Y}_k|\mathbf{X}_k) &= p(\mathbf{y}_k, \mathbf{Y}_{k-1}|\mathbf{x}_k, \mathbf{X}_{k-1}) \\ p(\mathbf{Y}_k|\mathbf{X}_k) &= p(\mathbf{y}_k|\mathbf{Y}_{k-1}, \mathbf{x}_k, \mathbf{X}_{k-1}) p(\mathbf{Y}_{k-1}|\mathbf{x}_k, \mathbf{X}_{k-1}) \\ p(\mathbf{Y}_k|\mathbf{X}_k) &= p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{Y}_{k-1}|\mathbf{X}_{k-1}), \end{aligned} \tag{2.10}$$

where the observations are assumed to be not conditional on the state in the future (Candy, 2016). The full prior distribution $p(\mathbf{X}_k)$ can be decomposed via Bayes' rule as:

$$p(\mathbf{X}_k) = p(\mathbf{x}_k|\mathbf{X}_{k-1}) p(\mathbf{X}_{k-1}), \tag{2.11}$$

where $p(\mathbf{x}_k|\mathbf{X}_{k-1})$ is the state evolution distribution. We can also decompose the evidence $p(\mathbf{Y}_k)$ via Bayes' rule as:

$$p(\mathbf{Y}_k) = p(\mathbf{y}_k|\mathbf{Y}_{k-1}) p(\mathbf{Y}_{k-1}). \tag{2.12}$$

Finally, according to Equations 2.10 – 2.12, we can rewrite Equation 2.9 in the recursive form as:

$$p(\mathbf{X}_k|\mathbf{Y}_k) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{Y}_{k-1}|\mathbf{X}_{k-1})p(\mathbf{x}_k|\mathbf{X}_{k-1})p(\mathbf{X}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})p(\mathbf{Y}_{k-1})}$$

$$p(\mathbf{X}_k|\mathbf{Y}_k) = p(\mathbf{X}_{k-1}|\mathbf{Y}_{k-1})\frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{X}_{k-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})}, \tag{2.13}$$

where $p(\mathbf{X}_{k-1}|\mathbf{Y}_{k-1})$ is the previous posterior PDF for $k \in \{2, \dots\}$. If $k = 1$,

$$p(\mathbf{x}_1|\mathbf{y}_1) = p(\mathbf{x}_0)\frac{p(\mathbf{y}_1|\mathbf{x}_1)p(\mathbf{x}_1|\mathbf{x}_0)}{p(\mathbf{y}_1)}, \tag{2.14}$$

where $p(\mathbf{x}_0) = p(\mathbf{x}_0|\mathbf{y}_0)$ is the initial prior PDF because observation $\mathbf{y}_0$ does not exist (Candy, 2016). We can also rewrite Equation 2.14 as:

$$p(\mathbf{X}_k|\mathbf{Y}_k) = p(\mathbf{x}_0)\frac{\prod_{m=1}^{k} p(\mathbf{y}_m|\mathbf{x}_m)p(\mathbf{x}_m|\mathbf{X}_{m-1})}{p(\mathbf{y}_k|\mathbf{Y}_{k-1})}. \tag{2.15}$$

2.2 Particle Filtering

Any posterior PDFs are continuous which cause Bayesian approaches impractical to implement computing devices. Particle filtering (PF) is a sequential Monte Carlo (SMC) method which generates samples of hidden states (called particles) to approximate the posterior PDF as:

$$p(\mathbf{X}_k|\mathbf{Y}_k) \approx \sum_{i=1}^{N} \widehat{w}_{0:k}^{i}\delta(\mathbf{X}_k - \mathbf{X}_k^i), \tag{2.16}$$

where $\widehat{w}_{0:k}^{i}$ is the probability value of the i-th sample state matrix at time step k, $\mathbf{X}_k^i$. Quantity N is the number of particles set by the user and $\delta(\cdot)$ is the Dirac delta function (Candy, 2016; Ristic et al., 2004). That is, $\widehat{w}_{0:k}^{i}$ denotes the normalized importance weight of particle $\mathbf{X}_k^i$ as:

$$\widehat{w}_{0:k}^{i} = \frac{w_{0:k}^{j}}{\sum_{j=1}^{N} w_{0:k}^{j}}, \tag{2.17}$$

where $w_{0:k}^{i}$ represents the true importance weight of particle $\mathbf{X}_k^i$ that can be found from

$$w_{0:k}^{i} \propto \frac{p(\mathbf{X}_k^i|\mathbf{Y}_k)}{q(\mathbf{X}_k^i|\mathbf{Y}_k)}, \tag{2.18}$$

where $q(\mathbf{X}_k|\mathbf{Y}_k)$ is the proposal distribution that draws particles $\mathbf{X}_k^i$ because we cannot directly draw particles from the true posterior PDF $p(\mathbf{X}_k|\mathbf{Y}_k)$. The approximated posterior PDF gets closer to the true posterior PDF as N increases (Candy, 2016; Ristic et al., 2004).

2.2.1 Sequential Importance Sampling

In case states and observations are sequential, we require particles that approximate the posterior PDF at a previous time step. In other words, particle weights need to be updated for each time step. Sequential importance sampling (SIS) is an algorithm derived from the concept of particle filtering for such a case (Candy, 2016; Ristic et al., 2004). The marginal importance distribution which is employed to draw particles $\mathbf{X}_k^i$ can be obtained by decomposing the importance distribution $q(\mathbf{X}_k|\mathbf{Y}_k)$ with Bayes' rule as:

$$q(\mathbf{X}_k|\mathbf{Y}_k) = q(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k)q(\mathbf{X}_{k-1}|\mathbf{Y}_{k-1}), \tag{2.19}$$

where $q(\mathbf{x}_k|\mathbf{X}_{k-1}, \mathbf{Y}_k)$ is the proposal distribution employed to draw sample vector $\mathbf{x}_k^i$ which can be added to sample matrix $\mathbf{X}_{k-1}^i$ to obtain particle $\mathbf{X}_k^i$. According to Equations 2.13 and 2.19, we can rewrite Equation 2.18 to express the recursive weight updating equation as:

$$w_{0:k}^{i} \propto \frac{p(\mathbf{X}_{k-1}^i|\mathbf{Y}_{k-1})p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{X}_{k-1}^i)}{q(\mathbf{X}_{k-1}^i|\mathbf{Y}_{k-1})q(\mathbf{x}_k^i|\mathbf{X}_{k-1}^i,\mathbf{Y}_k)p(\mathbf{y}_k|\mathbf{Y}_{k-1})}$$

$$w_{0:k}^{i} \propto w_{0:k-1}^{i} \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{X}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{X}_{k-1}^i,\mathbf{Y}_k)}, \tag{2.20}$$

where $w_0^i = 1/N$ is the initial weight assigned to all initial particles that are drawn from the initial proposal distribution $q(\mathbf{x}_0)$. We discard term $1/[p(\mathbf{y}_k|\mathbf{Y}_{k-1})]$ and express $w_{0:k}^i$ in proportionality because all N particles are conditional on the same set of observations $\mathbf{Y}_k$ (Candy, 2016). First-order Markovian systems are systems where state $\mathbf{x}_k$ is conditional on only $\mathbf{x}_{k-1}$. Thus, sets $\mathbf{X}_{k-1}$ and $\mathbf{Y}_{k-1}$ are not necessary (Ristic et

al., 2004). If we need to estimate only state $\mathbf{x}_k$ of such a system, we can reduce Equation 2.20 to:

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i,\mathbf{y}_k)}, \tag{2.21}$$

where w_k^i denotes the true non-normalized weight of the sample state vector $\mathbf{x}_k^i$ (Ristic et al., 2004). We can also reduce Equation 2.16 in case we draw particles $\mathbf{x}_k^i$ instead of $\mathbf{X}_k^i$ as:

$$p(\mathbf{x}_k|\mathbf{Y}_k) \approx \sum_{i=1}^N \widehat{w}_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i), \tag{2.22}$$

where the state $\mathbf{x}_k$ can be computed with MAP estimation or with a weighted mean (WM) expressed as[3]:

$$\hat{\mathbf{x}}_k^{MAP} = \underset{\mathbf{x}_k^i}{\operatorname{argmax}}\, p(\mathbf{x}_k^i|\mathbf{Y}_k), \tag{2.23}$$

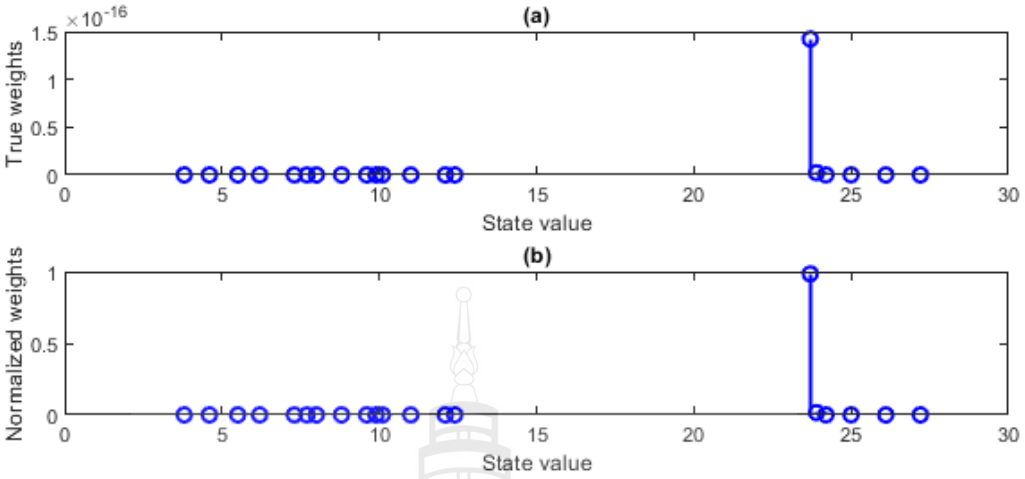
and

$$\hat{\mathbf{x}}_k^{WM} = \sum_{i=1}^N \widehat{w}_k^i \mathbf{x}_k^i. \tag{2.24}$$

The variance of particle weights grows as time increased, even after a few time steps. Particle degeneracy can then occur where only few particles have substantial weights while those of the other particles tend to zero as shown in Figure 2.2. Particle degeneracy causes poor performance in the posterior PDF approximation and state estimation. In the worst case, there can be only one particle with a non-zero weight (or unity normalized weight) (Candy, 2016; Ristic et al., 2004). The variance of particle weights can be found as:

$$\mathrm{Var}\left(w_k^i\right) = \left(w_{k-1}^i\right)^2 \left[\int \frac{p^2(\mathbf{y}_k|\mathbf{x}_k)p^2(\mathbf{x}_k|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k|\mathbf{x}_{k-1}^i,\mathbf{y}_k)} d\mathbf{x}_k - p^2(\mathbf{y}_k|\mathbf{x}_{k-1}^i)\right], \tag{2.25}$$

where $q(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)$ represents the proposal distribution which can be chosen by the user (Candy, 2016; Doucet, 2000; Ristic et al., 2004).



Note (a) A particle swarm of low-weight particles with true weights

(b) A particle swarm of low-weight particles with normalized weights

Figure 2.2 Low-weight particles with severe degeneracy

To obtain the minimum variance of weights as $\mathrm{Var}(w_k^i) = 0$, we draw particles from the proposal distribution $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i, \mathbf{y}_k)$ (Candy, 2016; Doucet, 2000). However, this PDF cannot be obtained directly because state $\mathbf{x}_k$ is not only conditional on the latest observation but also the previous state. Furthermore, it causes weight updating in Equation 2.20 to be expressed as:

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i,\mathbf{y}_k)} = w_{k-1}^i \frac{p(\mathbf{y}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{X}_{k-1}^i)p(\mathbf{y}_k|\mathbf{x}_{k-1}^i)}{p(\mathbf{y}_k|\mathbf{x}_k^i,\mathbf{x}_{k-1}^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)p(\mathbf{x}_{k-1}^i)}$$

$$w_k^i \propto w_{k-1}^i p(\mathbf{y}_k|\mathbf{x}_{k-1}^i), \tag{2.26}$$

where

$$p(\mathbf{y}_k|\mathbf{x}_{k-1}^i) = \int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)d\mathbf{x}_k, \tag{2.27}$$

which requires evaluating an integral and is, therefore, impractical (Doucet, 2000; Ristic et al., 2004). Note that $p(\mathbf{x}_{k-1}^i) = 1/N$ for $i \in \{1, \dots, N\}$ is a constant according to the assumption of "perfect sampling" and it can also be canceled out because we express Equation 2.26 in proportionality (Candy, 2016).

For simplicity, we prefer to draw each new particle $\mathbf{x}_k^i$ from the state evolution PDF $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$ (Candy, 2016). We can then approximate the state prediction PDF

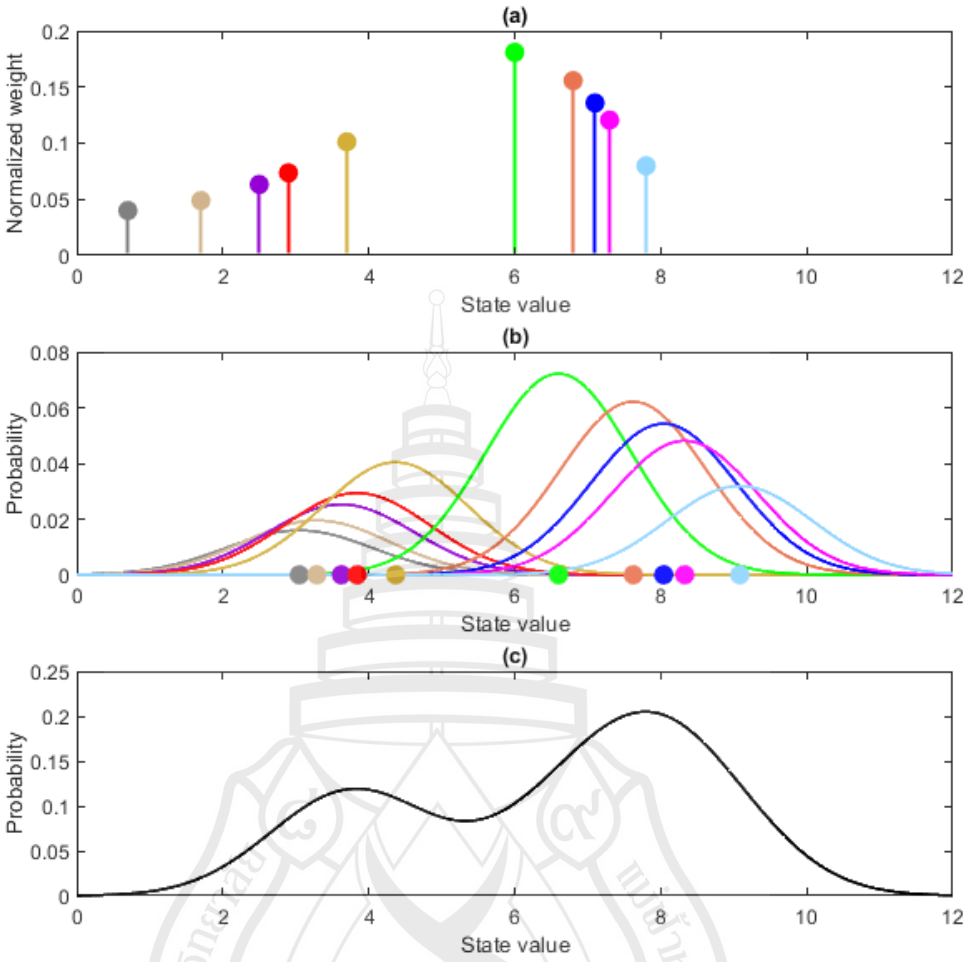
$p(\mathbf{x}_k|\mathbf{Y}_{k-1})$ by modifying the Chapman-Kolmogorov equation that was shown in Equation 2.4 as:

$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{Y}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1}) d\mathbf{x}_{k-1}$$
$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \approx \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) \sum_{i=1}^{N} \widehat{w}_{k-1}^{i} \delta(\mathbf{x}_{k-1} - \mathbf{x}_{k-1}^{i})\, d\mathbf{x}_{k-1}$$
$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \approx \sum_{i=1}^{N} \widehat{w}_{k-1}^{i} p(\mathbf{x}_k|\mathbf{x}_{k-1}^{i}), \tag{2.28}$$

which is the summation of N weighted state evolution PDFs as shown in Figure 2.3 where the state evolution function is assumed to be non-linear but with Gaussian noise, for simplicity. To be accurate, we select the weighted state evolution PDF $\widehat{w}_{k-1}^{i} p(\mathbf{x}_k|\mathbf{x}_{k-1}^{i})$ as the proposal distribution and this simplifies the weight calculation in Equation 2.20 as:

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{y}_k|\mathbf{x}_k^i) p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{\widehat{w}_{k-1}^i p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}$$
$$w_k^i \propto p(\mathbf{y}_k|\mathbf{x}_k^i), \tag{2.29}$$

because $w_{k-1}^i / \widehat{w}_{k-1}^i$ is the constant obtained from weight normalization. Choosing the weighted state evolution PDF $\widehat{w}_{k-1}^{i} p(\mathbf{x}_k|\mathbf{x}_{k-1}^{i})$ as the proposal distribution means that each PDF $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{i})$ is expected to draw $N\widehat{w}_{k-1}^{i}$ new prediction particles $\mathbf{x}_k^i$ at the beginning of time step k. That is, state values of high-weight particles are employed to form state evolution PDFs that are allowed to draw new particles in greater numbers at the next time step (Candy, 2016; Kuptametee & Aunsri, 2022a; Li et al., 2015). However, quantity $N\widehat{w}_{k-1}^{i}$ can be any non-negative, non-integer value. We can then force each state evolution PDF $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{i})$ to draw only one new particle $\mathbf{x}_k^i$ as shown in Figure 2.4 by setting $w_{k-1}^i = 1/N$ and $\widehat{w}_{k-1}^i = 1/N$ for every PDF $p(\mathbf{x}_k|\mathbf{x}_{k-1}^{i})$ in order to satisfy Equation 2.29 (Kuptametee & Aunsri, 2023).

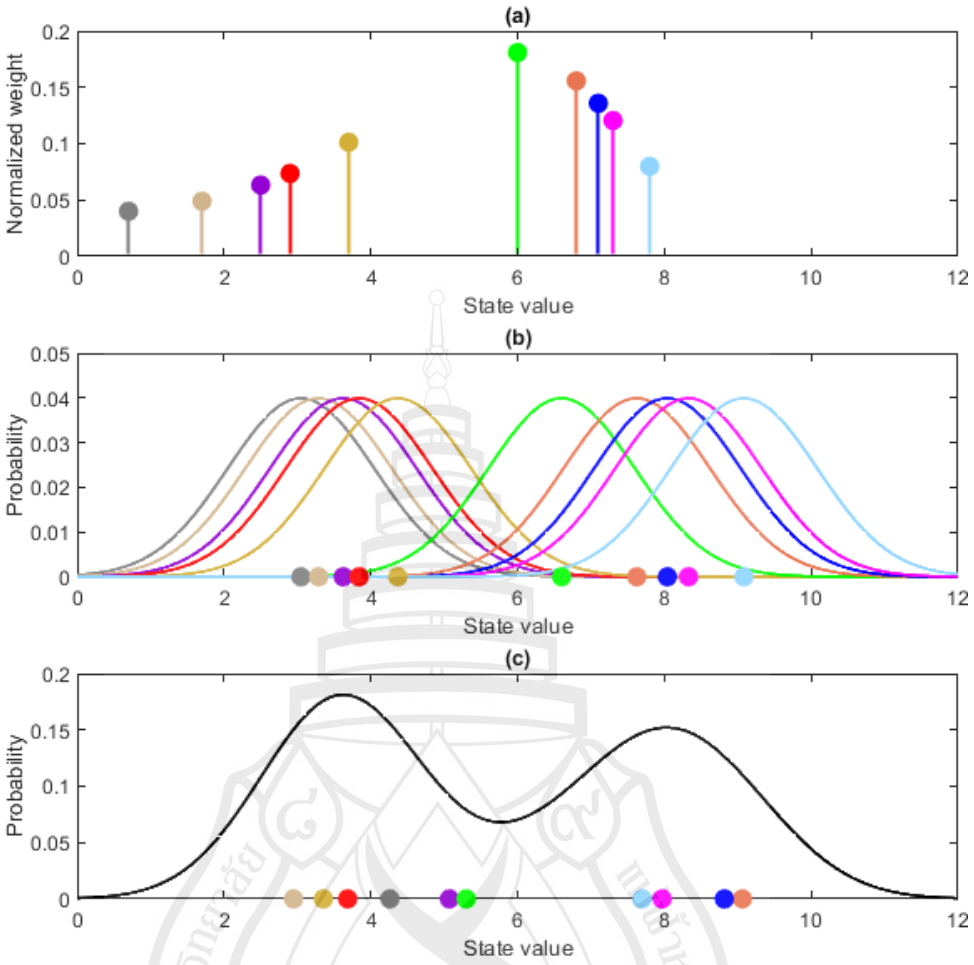


Note (a) An approximated posterior PDF $p(x_{k-1}|Y_{k-1})$

(b) Weighted proposal PDFs with new mean values $f_{k-1}(x_{k-1}^{i})$

(c) Theoretical state prediction PDF $p(x_k|Y_{k-1})$

Figure 2.3 A theoretical process of state prediction in SIS



Note (a) An approximated posterior PDF $p(x_{k-1}|Y_{k-1})$

(b) Equally weighted proposal PDFs with new mean values $f_{k-1}(x_{k-1}^i)$

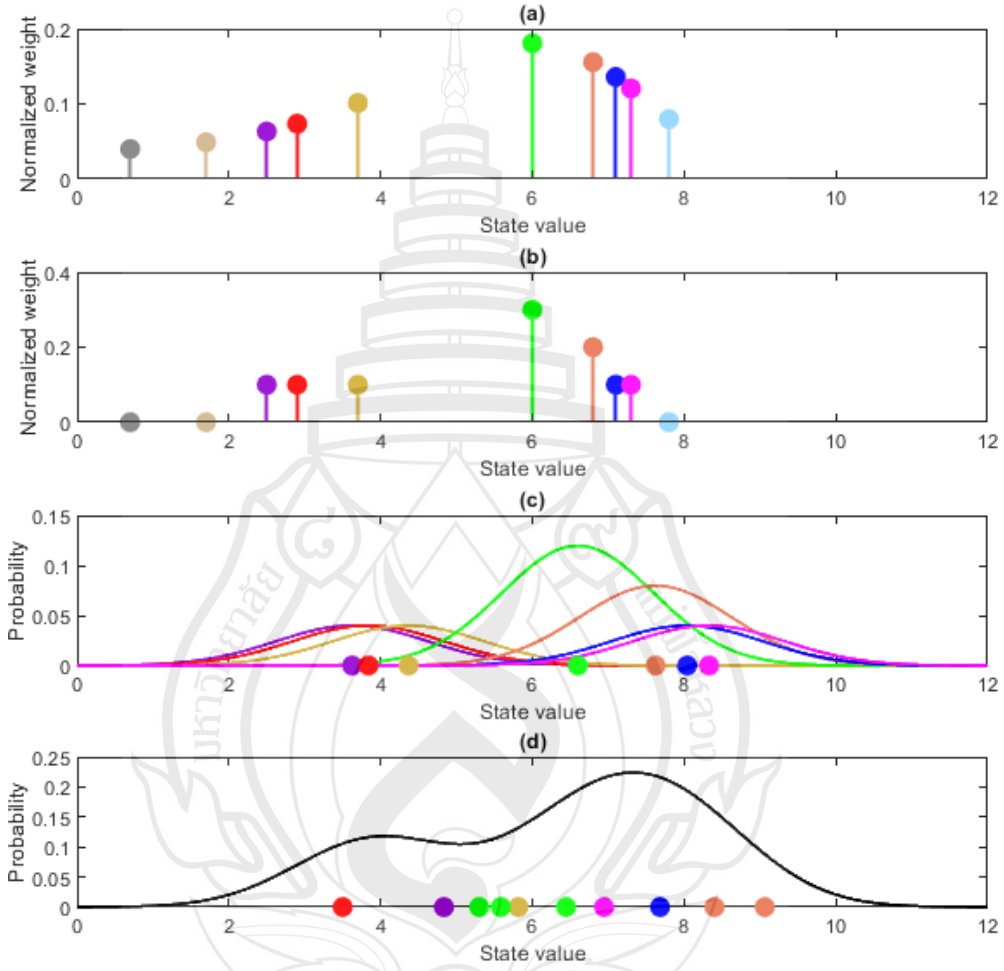
(c) A created state prediction PDF $p(x_k|Y_{k-1})$ and newly drawn particles x_k^i

Figure 2.4 A practical process of state evolution in SIS

2.2.2 Particle Resampling

As previously discussed, quantity $N\widehat{w}_{k-1}^i$, the expected number of new prediction particles to be drawn from each state evolution PDF $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$, is not always an integer. Thus, we employ a random selection method called "resampling" to draw new particles (Kuptametee & Aunsri, 2022a; Li et al., 2015). High-weight particles are more likely to be selected multiple times, while low-weight particles are more likely to not be selected. Consequently, the particle swarm will consist of replicas of high-weight particles and low-weight particles will be eliminated or reduced. Also, each particle weight must be reset to $1/N$ because these particles are no longer i.i.d.

and their state values will be employed to construct the new prior PDF that draws new particles that belong to the next time step (Candy, 2016). The state evolution of resampled particles is shown in Figure 2.5. For example, the normalized weight of the green particle is 0.3 when $N = 10$. Three green proposal PDFs that are weighted by term $1/N$ can then be summed and three green new particles can be drawn.



Note (a) An approximated posterior PDF $p(x_{k-1}|Y_{k-1})$

(b) A swarm of resampled particles $\tilde{x}_{k-1}^{i}$

(c) Weighted proposal PDFs with new mean values $f_{k-1}(\tilde{x}_{k-1}^{i})$

(d) A created state prediction PDF $p(x_k|Y_{k-1})$ and newly drawn particles x_k^i

Figure 2.5 State evolution of resampled particles

Multinomial resampling is the most basic scheme that employs the roulette wheel selection (RWS) algorithm to resample the particles (Larose, 2006; Ristic et al.,

2004). First, we find the cumulative distribution function (CDF) of the normalized weights of particles $\mathbf{x}_{k-1}^{i}$ as:

$$\widehat{CW}_{k-1}^{i} = \sum_{n=1}^{i} \widehat{w}_{k-1}^{n}, \tag{2.30}$$

where $\widehat{CW}_{k-1}^{0} = 0$ and $\widehat{CW}_{k-1}^{N} = 1$. The state values of the i-th particle $\mathbf{x}_{k-1}^{i}$ are then assigned to the j-th resampled particle $\tilde{\mathbf{x}}_{k-1}^{j}$ as:

$$\tilde{\mathbf{x}}_{k-1}^{j} = \mathbf{x}_{k-1}^{i}, \text{ if } \widehat{CW}_{k-1}^{i-1} < u_{k-1}^{j} \leq \widehat{CW}_{k-1}^{i}, \tag{2.31}$$

where u_{k-1}^{j} is a random value employed to find the resampled particle $\tilde{\mathbf{x}}_{k-1}^{j}$. It is drawn as $u_{k}^{j} \sim U(0,1)$ where each real number between 0 to 1 excluding the bounds has uniform probability to be drawn (Candy, 2016; Ristic et al., 2004). Then, we draw only one new particle $\mathbf{x}_{k}^{i}$ from each state evolution PDF $p(\mathbf{x}_{k}|\tilde{\mathbf{x}}_{k-1}^{i})$ as previously shown in Figure 2.5. If the normalized weight of particle $\mathbf{x}_{k-1}^{i}$ is high, the particle will have a high probability to be selected because the difference $\widehat{CW}_{k-1}^{i} - \widehat{CW}_{k-1}^{i-1}$ (which represents a part of a roulette wheel) is high.

Carpenter et al. (1999) proposed the systematic resampling that employs the stochastic universal sampling (SUS) algorithm (which was proposed by Baker (1987)) as an alternative to the RWS to reduce the selection bias (Ristic et al., 2004). The random value u_{k-1}^{j} employed in Equation 2.31 for this scheme is also modified as:

$$u_{k-1}^{j} = u_{k-1}^{1} + \frac{j-1}{N}, \tag{2.32}$$

where $u_{k-1}^{1} \sim U(0, 1/N)$ is the only one freely drawn random value. That is, systematic resampling is a quasi-random resampling scheme (Kuptametee & Aunsri, 2022a; Li et al., 2015). Figure 2.6 presents a pseudocode for employing RWS and SUS algorithms.

Input: N particles ($\mathbf{x}_{k-1}^{i}$) and their normalized weights ($\widehat{w}_{k-1}^{i}$)

Output: N resampled particles ($\tilde{\mathbf{x}}_{k-1}^{i}$)

$\widehat{CW}_{k-1}^{1} \leftarrow \widehat{w}_{k-1}^{1}$ %Initialize the CDF of N normalized weights

for $i \in \{2, \dots, N\}$ **do**

$\widehat{CW}_{k-1}^{i} \leftarrow \widehat{CW}_{k-1}^{i-1} + \widehat{w}_{k-1}^{i}$ %Find the next value of the CDF

end for

If SUS is employed **then**

$u_1 \sim U(0, 1/N)$ %A randomly drawn first resampling point

end if

for $j \in \{1, \dots, N\}$ **do**

If RWS is employed **then**

$u_j \sim U(0, 1)$ %Draw each resampling point

end if

If SUS is employed **then**

$u_j \leftarrow u_1 + (j-1)/N$ %Find the next resampling point

end if

$a \leftarrow 1$ %Start from the first value of the CDF

while $u_j > \widehat{CW}_{k-1}^{a}$ **do**

$a \leftarrow a + 1$ %Move to the next value of the CDF

end while

$\tilde{\mathbf{x}}_{k-1}^{j} \leftarrow \mathbf{x}_{k-1}^{a}$ % j-th resampled particle

end for

Figure 2.6 A pseudocode for RWS and SUS algorithms

Both RWS and SUS consume substantial computation time because they employ values of the CDF of all N normalized weights (found via Equation 2.30) to select each particle sequentially as shown in Equation 2.31. Li et al. (2013) then proposed the rounding-copy resampling that allows each of N original particles $\mathbf{x}_{k-1}^{i}$ to create $[N\widehat{w}_{k-1}^{i}]$ replicas where $[\cdot]$ is a rounding symbol (e.g., $[8.4] = 8$ and $[8.5] = 9$) and $\widehat{w}_{k-1}^{i}$ is the normalized weight of each particle $\mathbf{x}_{k-1}^{i}$. While this scheme does not require the CDF of all N normalized weights, the total number of particles obtained after resampling may not be N. In case the total number of resampled particles is greater

than N, we keep only the first N best particles and we then reset the weights of these particles to $1/N$. If the total number of resampled particles is less than N, we must create more replicas of some resampled particles in order to obtain N resampled particles in total.

Aunsri et al. (2021) proposed the percentile-based resampling that modifies rounding-copy resampling by allowing only high-weight particles to replicate themselves while the others are eliminated from the swarm. After all N original particles $\mathbf{x}_{k-1}^{i}$ are sorted by their weights in descending order, only the first $N_{PBR,k-1}$ particles are kept while the others are eliminated from the swarm. Quantity $N_{PBR,k-1}$ is the smallest number of particles that satisfies the condition that summation of the weights of the surviving $N_{PBR,k-1}$ particles must not be less than a preset percentage of the summation of weights of the N original particles $\mathbf{x}_{k-1}^{i}$. That is, $N_{PBR,k-1} \leq N$ and we must normalize weights of the surviving $N_{PBR,k-1}$ particles as:

$$\widehat{w}_{k-1,dsc}^{i} = \frac{w_{k-1,dsc}^{i}}{\sum_{j=1}^{N_{PBR,k-1}} w_{k-1,dsc}^{j}}, \tag{2.33}$$

where $\widehat{w}_{k-1,dsc}^{i}$ denote the weights of the i-th sorted particles and $i \in \{1, \ldots, N_{PBR,k-1}\}$. The number of replicas of each surviving particle is suggested to be $\lceil N\widehat{w}_{k-1,dsc}^{i} \rceil$ where $\lceil \cdot \rceil$ is the ceiling symbol (e.g., $\lceil 8.1 \rceil = 9$). This ensures that the total number of particles obtained after resampling can only be equal to or greater than N (Aunsri et al., 2021). More resampling schemes are reviewed and discussed by Kuptametee and Aunsri (2022a) and Li et al. (2015).

The main side effect of employing particle resampling is that the state values of particles will have reduced diversity. This problem is called particle impoverishment where the particle swarm converges to only one or just a few state values and the opportunity to discover high-weight state values decreases. This problem can be severe if the variance of the state evolution noise or the width of each weighted proposal PDF in Figure 2.5(c) is too small. Consequently, the particle swarm may not be able to discover high-weight state values effectively at the next time steps (Candy, 2016; Kuptametee et al., 2024; Ristic et al., 2004).

The effective sample size (ESS) is employed to measure particle degeneracy and to decide whether or not resampling should be employed. The ESS at time step $k-1$ can be calculated as:

$$ESS_{k-1} = \frac{N}{1+\mathrm{Var}(w_{k-1}^{i})}, \tag{2.34}$$

where $1 \leq ESS_{k-1} \leq N$ and a low ESS_{k-1} denotes highly severe particle degeneracy where only a few particles have high weights (Candy, 2016; Martino et al., 2017; Ristic et al., 2004). That is, we may resample particles when ESS_{k-1} is lower than a preset threshold. As shown in Equation 2.25, $\mathrm{Var}(w_{k-1}^{i})$ cannot be found easily because it requires an integral evaluation. Quantity ESS_{k-1} can then be approximated as:

$$ESS_{k-1} \approx \frac{\left(\sum_{i=1}^{N} w_{k-1}^{i}\right)^{2}}{\sum_{i=1}^{N}\left(w_{k-1}^{i}\right)^{2}} = \frac{1}{\sum_{i=1}^{N}\left(\widetilde{w}_{k-1}^{i}\right)^{2}}, \tag{2.35}$$

which can be found from either the normalized weights or the true non-normalized weights (Ahwiadi & Wang, 2020; Kuptametee & Aunsri, 2022a; Ristic et al., 2004).

Some particles are considered as "high-weight particles" because the weights of the others are significantly lower or are near-zero, but their true non-normalized weights may be low, as previously shown in Figure 2.2. That is, the local maximum state values (i.e., the best particles that we have on hand) may have low true weights. In a particle swarm with maximum ESS (i.e., $ESS_{k-1} = N$), all true weights w_{k-1}^{i} are equal but can be low, while their state values are not necessarily the same. That is, a high ESS_{k-1} value does not mean that the hidden state will be estimated effectively (Kuptametee et al., 2024).

Sequential importance resampling (SIR) always performs particle resampling to eliminate low-weight particles at every time step regardless of the ESS value (Arulampalam et al., 2002; Gordon et al., 1993; Ristic et al., 2004). To fix the impoverishment that can occur after resampling is employed, we can perturb the state values of the resampled particles to regain particle diversity. In other words, we need to find completely new state vectors that are different from the local maximum state vectors. Weights of the new state vectors are also expected to be higher than the original

non-reset weights of the resampled particles (Candy, 2016; Kuptametee & Aunsri, 2022a; Ristic, 2004).

Pitt and Shephard (1999) proposed the auxiliary particle filter that draws auxiliary particles that assist in finding the high-weight state vectors at the next time step. For simplicity, suppose that the state evolution function is a Gaussian PDF:

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}) + \mathbf{u}_{k-1}, \tag{2.36}$$

where $\mathbf{u}_{k-1} \sim N(0, \mathbf{Q}_{k-1})$ and $\mathbf{Q}_{k-1}$ is a $d_u \times d_u$ covariance matrix of state evolution noise that is employed to find new state values at time step k. We employ particles $\mathbf{x}_{k-1}^i$ to find each new state vector $\mathbf{x}_{k,aux}^i = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}^i)$ as an auxiliary particle that contains mean values of the Gaussian state evolution PDF at time step k. We employ the likelihood value $p(\mathbf{y}_k|\mathbf{x}_{k,aux}^i)$ as the weight of each auxiliary particle $\mathbf{x}_{k,aux}^i$ and we can then resample these N auxiliary particles. Finally, we find each particle $\mathbf{x}_k^i$ by adding state evolution noise $\mathbf{u}_{k-1}^i$ to each resampled auxiliary particle $\tilde{\mathbf{x}}_{k,aux}^i$ (according to Equation 2.36) and we evaluate the weight of every particle $\mathbf{x}_k^i$ (according to Equation 2.29) to obtain the approximated posterior PDF at time step k (Pitt & Shephard, 1999). However, the variance values of covariance matrix $\mathbf{Q}_{k-1}$ must be carefully set. If the variance values of $\mathbf{Q}_{k-1}$ are too high, particles $\mathbf{x}_k^i$ may be blindly located at regions of state values that have low weights. On the other hand, if variance values of $\mathbf{Q}_{k-1}$ are too low, the diversity of state vectors may still be low and particle impoverishment may not be properly remedied (Kuptametee & Aunsri, 2022a; Ristic, 2004).

Musso et al. (2001) proposed the regularized particle filter to regain the diverisity of the post-resampling particles. Regularized particle filter employs state values of each resampled particle $\tilde{\mathbf{x}}_{k-1}^i$ to construct a symmetric continuous kernel function that is centered at state values of $\tilde{\mathbf{x}}_{k-1}^i$. Then, we draw each new particle $\mathbf{x}_{k-1,reg}^i$ from each created i-th kernel function to obtain a new set of N particles with regained diversity. Finally, we evaluate the weight of each new particle $\mathbf{x}_{k-1,reg}^i$ as the likelihood value $p(\mathbf{y}_{k-1}|\mathbf{x}_{k-1,reg}^i)$ to obtain the new approximated posterior PDF at the same time step $k-1$ (Musso et al., 2001). There are many choices of kernel functions that can be employed, for example, Epanechnikov, box (or uniform), Gaussian and

triangle (Candy, 2016; Gordon et al., 1993; Kuptametee et al., 2024). However, kernel functions must be designed carefully in order to ensure that particle impoverishment will be properly addressed and new particles $\mathbf{x}_{k-1,reg}^{i}$ will not be located at regions of low-weight state values.

The roughening scheme perturbs state values of each of N resampled particles $\tilde{\mathbf{x}}_{k-1}^{i}$ at time step $k-1$ by adding zero-mean Gaussian random values that are drawn from the diagonal covariance matrix $\boldsymbol{\Sigma}_{k-1} = \text{diag}(\sigma_{k-1,1}^{2}, \ldots, \sigma_{k-1,d_x}^{2})$ (Gordon et al., 1993). Each variance value $\sigma_{k-1,m}^{2}$ for perturbing the state value of the m-th vector component where $m \in \{1, \ldots, d_x\}$ is found as:

$$\sigma_{k-1,m}^{2} = \left[\beta\left(\tilde{x}_{k-1,m,max} - \tilde{x}_{k-1,m,min}\right)N^{-\frac{1}{d_x}}\right]^{2}, \tag{2.37}$$

where $\beta > 0$ is a preset tuning parameter. Quantities $\tilde{x}_{k-1,m,min}$ and $\tilde{x}_{k-1,m,max}$ are the minimum and maximum state values of the m-th vector component at time step $k-1$ that must be found from the swarm of N resampled particles (Gordon et al., 1993).

Han et al. (2015) proposed the adaptive fission particle filter (AFPF) that modifies rounding-copy resampling by diversifying state values of replicas of N original particles. Each j-th replica of the original particle $\mathbf{x}_{k-1}^{i}$ is drawn from a Gaussian PDF as:

$$\mathbf{x}_{k-1,rep(i)}^{j} \sim N(\mathbf{x}_{k-1}^{i}, \lambda_{k-1}^{i}\boldsymbol{\Sigma}), \tag{2.38}$$

where $j \in \{1, \ldots, \lfloor N\hat{w}_{k-1}^{i} \rfloor + N_{rep,min}\}$; $N_{rep,min} \geq 0$ the preset minimum number of replicas that will be created from each original particle, $\boldsymbol{\Sigma}$ is a symmetric $d_x \times d_x$ covariance matrix that is designed by the user, and

$$\lambda_{k-1}^{i} = \frac{1}{1+\exp\left(\frac{w_{k-1}^{i} - \text{Avg}(w_{k-1}^{i})}{\max(w_{k-1}^{i}) - \text{Avg}(w_{k-1}^{i})}\right)}, \tag{2.39}$$

represents a parameter called “fission factor” that is exclusively found for each original particle $\mathbf{x}_{k-1}^{i}$ (Han et al., 2015). Function $\exp(\cdot)$ denotes the exponential function.

Quantities $\max(w_{k-1}^i)$ and $\text{Avg}(w_{k-1}^i)$ are the maximum and average values of the non-normalized weights of N original particles $\mathbf{x}_{k-1}^i$, respectively. Parameter λ_{k-1}^i tunes the variance values of the covariance matrix $\mathbf{\Sigma}$ that is employed to perturb the state values of replicas of the original particle $\mathbf{x}_{k-1}^i$. If the weight w_{k-1}^i is high, λ_{k-1}^i will be low and replicas will be created in a great number and are located close to their parent. If the weight w_{k-1}^i is low, λ_{k-1}^i will be high and replicas will be created in a low number and are located away from their parent. The theoretical minimum value of λ_{k-1}^i in Equation 2.39 is at $1/[1 + \exp(1)]$, while its theoretical maximum value approaches 1 asymptotically. After every new replica is found, the N original particles are gathered along with all created replicas forming the new set of particles. Finally, weight sorting must be done to keep only the N best particles from this set that are then further employed in state estimation (Han et al., 2015).

2.3 Genetic Algorithms

Genetic Algorithms (GAs) are proposed by Holland (1992) as the methods employed to randomly find the best solution for optimization problems. GAs are inspired by the natural selection process where the fittest individuals have increased chances to survive. Then, the survived individuals produce new offspring individuals with high diversity and greater fitness. In offspring creation, a chromosome of a parent and that of another parent are employed to create a pair of new offspring chromosomes. However, mutation may occur where some genes of the offspring chromosomes alter.

In traditional GAs, we treat a binary string as a chromosome where bits (zeros and ones) denote genes. However, state values in practical applications are real numbers. These state values must first be converted into binary strings, where the number of bits must be carefully considered. Longer length mitigates loss of information that is due to conversions between real numbers and binary strings and vice versa, but a higher computational cost is required (Larose, 2006). We can alternatively employ arithmetic GAs which directly treat a vector of real number state values as a chromosome, while each real number state value of the vector represents a gene. Thus,

binary GAs are out of scope in this dissertation; more information about binary GAs can be found in work by Katoch et al. (2021), Larose (2006), and Michalewicz (1996).

After new offspring state vectors are produced, the state vector with the highest fitness value is chosen as the best solution for the problem. GAs can be employed in, for example, shortest path planning, localization, prices prediction, neural networks, and video processing (Alam et al., 2020; Drachal & Pawłowski, 2021; Michalewicz, 1996).

2.3.1 Parent Selection

High-fit individuals have high probabilities to survive the selection process. The most basic selection process is to employ the RWS algorithm which is also employed in multinomial resampling as discussed in Subsection 2.2.2. That is, particle weights found according to Equation 2.29 are employed as fitness values, while normalized weights can be employed as selection probabilities (Larose, 2006; Ristic et al., 2004). There are also other selection schemes that can be employed. Rank selection and tournament ranking involve competitions between individuals that are not necessarily based on the natural selection process (Katoch et al., 2021; Kuptametee et al., 2024; Larose, 2006). De Jong (1975) proposed the elitism technique which ensures that the state values of the individuals with the highest weights will appear in the next generation without being altered during offspring creation (Kuptametee et al., 2024; Larose, 2006).

If there is an individual that dominates one significantly large part of the selection wheel while the other parts are very small, the parents will consist of too many replicas of this high-fit individual and diversity of parents will be low (Larose, 2006). The Sigma scaling technique ensures diversity of individuals by modifying each fitness value of the i-th individual (i.e., particle $\mathbf{x}_k^i$) that will be employed in the selection process as:

$$w_{k,SS}^i = 1 + \frac{w_k^i - \mathrm{Avg}(w_k)}{\mathrm{Std}(w_k)}, \tag{2.40}$$

where w_k^i denotes the original fitness value of the i-th individual (or particle weight found according to Equation 2.29) and $\mathrm{Avg}(w_k)$ denotes the average of all N original fitness values (Katoch et al., 2021; Larose, 2006). If the standard deviation of all N

original fitness values, $\mathrm{Std}(w_k)$, is high, the sigma-scaled fitness values of unfit individuals will increase. Consequently, unfit individuals have a higher chance to be selected as parents. On the contrary, if $\mathrm{Std}(w_k)$ is low, high-fit individuals will gain greater dominance in offspring creation by having higher sigma-scaled fitness values. Note that we must first normalize the modified fitness values of every individual to obtain the modified selection probability values that can be employed in any selection schemes (Larose, 2006).

Boltzmann selection initially widens the exploration scope on the search space in order to promote diversity of individuals. Then, it narrows the search scope to make GA converge to the optimal solution more quickly at later generations (Katoch et al., 2021; Larose, 2006). That is, selection probabilities of unfit individuals are almost as high as those of high-fit individuals in the beginning generations. Then, selection probabilities of unfit individuals decrease at later generations (Katoch et al., 2021; Larose, 2006). The modified fitness values in Boltzmann selection can be found as:

$$w_{k,BS}^{i} = \frac{\exp(w_k^i/T)}{\mathrm{Avg}[\exp(w_k/T)]}, \tag{2.41}$$

where T is a parameter called "temperature" that is initially set to be high and decreases with time. If T decreases, the high-fit individuals are more likely to be selected as parents. Again, we must first normalize each modified fitness value to obtain the modified selection probability of each individual (Larose, 2006).

The effects of employing the modified fitness values found with the Sigma scaling and Boltzmann selection techniques can be significant if the number of generations is set to be high (Katoch et al., 2021; Larose, 2006). However, the diversity of individuals and the number of generations must be carefully considered and be balanced with computation time. Also, recall that the modified fitness values can only be employed in the selection process. We must employ the original fitness values to consider the optimality of each individual to be the best solution of the optimization problem (Kuptametee et al., 2024).

2.3.2 Offspring Creation

Suppose that there are N individuals that survive and are included in the set of parents. We can randomly create up to $\lfloor N/2 \rfloor$ pairs of parents where $\lfloor \cdot \rfloor$ is the floor

symbol (e.g., $\lfloor 9.9 \rfloor = 9$). When the quantity N is odd, the number of individuals that skip the offspring creation (or do not pair with any other individuals) must also be odd and be at least one. These skipped individuals should have the highest fitness values according to the principle of the elitism technique and their state values definitely appear in the next generation of the population (De Jong, 1975). There are two steps in calculating new state values of the to-be-created offspring vectors, crossover and mutation (Katoch et al., 2021; Larose, 2006; Michalewicz, 1996).

In the crossover process, two parent chromosomes exchange some of their genes in order to produce two new offspring chromosomes. Discrete crossover follows the principle of natural offspring chromosome creation (Larose, 2006). Suppose that there are two parent vectors with length of five state values, $\{0.4, 0.7, 0.5, 0.8, 1.1\}$ and $\{0.6, 0.3, 1.2, 1.0, 0.9\}$. For example, the first state value of the first offspring vector can be either 0.4 or 0.6 with equal probability. If the first offspring vector is $\{0.6, 0.7, 0.5, 1.0, 1.1\}$, another offspring vector must then be $\{0.4, 0.3, 1.2, 0.8, 0.9\}$. However, the number of possible patterns of offspring vectors in this example is only $2^5 = 32$. That is, the diversity of individuals can still be limited (Larose, 2006).

Arithmetic crossover creates two offspring vectors with state values that can be completely different from those of their two parents in order to ensure diversity (Larose, 2006). Suppose that there are two parent vectors, $\mathbf{x}_k^a$ and $\mathbf{x}_k^b$, where $a \in \{1, \dots, N\}$ and $b \in \{1, \dots, N\}$ but $a \neq b$. Two new offspring vectors can be found as:

$$\mathbf{x}_k^{a,off} = \begin{cases} \alpha \mathbf{x}_k^b + (1-\alpha)\mathbf{x}_k^a, & u \leq p_c \\ \mathbf{x}_k^a, & u > p_c \end{cases} \tag{2.42a}$$

$$\mathbf{x}_k^{b,off} = \begin{cases} \alpha \mathbf{x}_k^a + (1-\alpha)\mathbf{x}_k^b, & u \leq p_c \\ \mathbf{x}_k^b, & u > p_c, \end{cases} \tag{2.42b}$$

where $\alpha \sim U(0,1)$ is a uniform random value that tunes the state values of the two offspring vectors. Parameter p_c represents the probability of crossover (p_c) where $0 \leq p_c \leq 1$ and $u \sim U(0,1)$. Parameter p_c is normally set to be high because individuals are encouraged to create new offspring (Katoch et al., 2021; Larose, 2006; Michalewicz, 1996). That is, the expected number of pairs that will create new offspring is $p_c \times \lfloor N/2 \rfloor$ pairs.

Parameter α can be independently drawn for each pair of two parents. When each parent is a one-dimensional (1-D) state value, the two linear equations for finding new offspring state values cross each other at $\alpha = 0.5$ as shown in Figure 2.7. Circles denote state values of two new offspring according to the parameter α (shown as a magenta dashed line). State values of the two offspring vectors are located between those of their two parents and are also located within bounds (i.e., minimum and maximum state values) of the population (Larose, 2006).

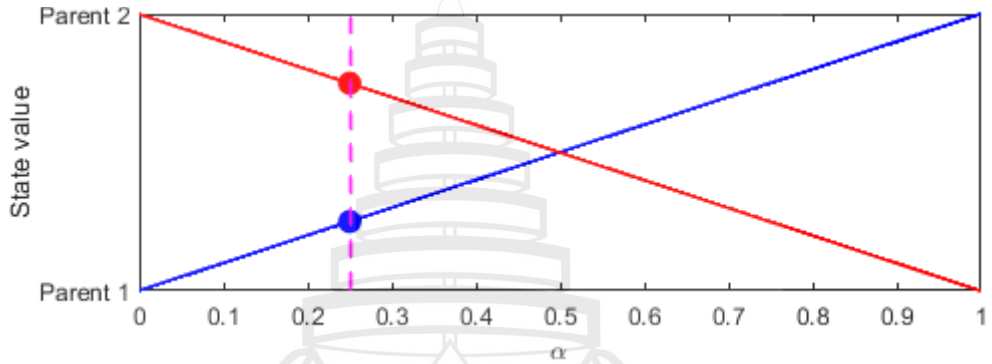


Figure 2.7 Two new offspring state values found via arithmetic crossover

If two parent vectors are identical, the two new offspring vectors that are created by employing arithmetic crossover will be exact copies of their parents. Radcliffe (1990) proposed the flat crossover that creates only one new offspring to save computation time. First, we need to compare fitness values of the two parents. Then, the one created offspring replaces the less-fit parent, while the fitter parent is kept unchanged. Suppose that there are two parent vectors, a high-fit parent $\mathbf{x}_k^{high}$ and an unfit parent $\mathbf{x}_k^{low}$. We can then find the offspring vector as:

$$\mathbf{x}_k^{off} = \alpha \mathbf{x}_k^{high} + (1 - \alpha)\mathbf{x}_k^{low}, \tag{2.43}$$

where a large value of parameter α tunes the state values of the offspring vector to become closer to those of its high-fit parent (Radcliffe, 1990).

Mutation ensures the diversity of individuals in the new generation by randomly altering some genes of the two offspring chromosomes created in the crossover step. In traditional GAs, each state value of all offspring vectors will be perturbed with the probability of mutation (p_m). That is, the expected total number of state values of the

whole offspring generation that will mutate after crossover is employed is $p_m \times N \times d_x$ values where $0 \leq p_m \leq 1$ and d_x represents length of state vector (Larose, 2006; Michalewicz, 1996). Parameter p_m can also be defined as the probability that all state values of an offspring vector will mutate, while the probability that none of state values of the offspring vector will mutate is $1 - p_m$ (Katoch et al., 2021). The reason is that, depending on the application, a state vector can consist of state values with different units. If some state values of such vectors are mutated while the rest are not, the optimization results can be highly erroneous because each new state value is inconsistently found. However, there can be cases where the mutated offspring vectors will contain abnormal state values that are located out of bounds. Also, any offspring state vector found by employing crossover may be replaced by its mutated replica with the lower weight. Thus, the parameter p_m should be set to a low value (Larose, 2006).

Gaussian mutation perturbs the offspring vectors by adding zero-mean Gaussian random values to offspring state values. In other words, the state values of the offspring vector are employed as mean values of the Gaussian PDF for drawing another new state vector (Larose, 2006). Suppose that a whole offspring vector $\boldsymbol{x}_k^{off}$ mutates. The new state vector can then be found as:

$$\mathbf{x}_k^{off\_m} \sim N\left(\mathbf{x}_k^{off}, \boldsymbol{\Sigma}\right), \text{ if } u \leq p_m, \tag{2.44}$$

where $\boldsymbol{\Sigma}$ is a $d_x \times d_x$ symmetric covariance matrix and $u \sim U(0,1)$. We can choose a diagonal matrix as the covariance matrix $\boldsymbol{\Sigma}$ when there are no correlations between each state value of the vector. However, variance values must be carefully set in order to prevent abnormally high or low state values (Kuptametee et al., 2024; Larose, 2006).

Uniform mutation can be employed in case the minimum and the maximum of the acceptable state values are known (Michalewicz, 1996). That is,

$$x_{k,m}^{off\_m} \sim U\left[x_{k,m}^{LB}, x_{k,m}^{UB}\right], \text{ if } u \leq p_m, \tag{2.45}$$

where $x_{k,m}^{off\_m}$ denotes the m-th vector component of the mutated offspring vector $\mathbf{x}_k^{off\_m}$, $u \sim U(0,1)$ and $m \in \{1, \ldots, d_x\}$. $U\left[x_{k,m}^{LB}, x_{k,m}^{UB}\right]$ denotes a uniform distribution where each real number between $x_{k,m}^{LB}$ and $x_{k,m}^{UB}$ (including both) can be drawn.

Quantities $x_{k,m}^{LB}$ and $x_{k,m}^{UB}$ denote the lower and upper bounds of the state values of the m-th vector component that must be set by the user, respectively (Michalewicz, 1996). While the new mutated state values are not located out of bounds, the original state values of the offspring vector $\mathbf{x}_k^{off}$ found in the crossover step can be destroyed. More crossover and mutation schemes are discussed and can be found in work by Katoch et al. (2021).

After we obtain new offspring individuals (where some of them mutate) at the desired number, we re-evaluate the fitness value of every individual in the most recent generation and repeat the overall GA process to find the next-generation individuals until our termination criteria are met. The termination criteria can be, for example, when a preset maximum number of generations (i.e., attempts of finding new high-fit state vectors) per time step is reached or when the average of fitness values of every individual no longer increases (Garzelli et al., 2008; Kuptametee et al., 2024; Larose, 2006).

2.4 Related Work

Resampling in generic PF algorithms replicates high-weight particles and eliminates low-weight particles. The resampled particles then are employed to predict the state at the next time step. The selection process in GAs also replicates high-fit individuals for creating new offspring values that belong to the same time step, while unfit individuals are eliminated. That is, particle resampling is technically similar to the selection process in GAs, while their objectives are different (Kuptametee et al., 2024).

Because parent selection reduces the diversity of the parent state vectors, Park et al. (2009) suggested that crossover and mutation should be employed to diversify the state values of N resampled particles $\tilde{\mathbf{x}}_{k-1}^i$ (i.e., replicas of selected parents) before entering the state evolution function. If the diversity of new particles $\tilde{\mathbf{x}}_{k-1,GA}^i$ was high, the diversity of the new mean values $\mathbf{f}_{k-1}\left(\tilde{\mathbf{x}}_{k-1,GA}^i\right)$ and that of the new prediction particles $\mathbf{x}_k^i$ would also be high. Consequently, there could be high likelihoods of discovering high-weight state vectors at time step k. However, recall that the true posterior PDF can be time-varying because both true state and observation data can

evolve with time. There could still be a chance that all of the new particles $\mathbf{x}_k^i$ would have low true weights.

Zhou et al. (2021) employed roughening on N resampled particles $\tilde{\mathbf{x}}_k^i$ (that are selected via the RWS algorithm) to obtain a set of N new parents $\tilde{\mathbf{x}}_{k,GA}^i$ with regained diversity. Also, the weight of each new parent $\tilde{\mathbf{x}}_{k,GA}^i$ had to be re-evaluated as the new likelihood value $p(\mathbf{y}_k|\tilde{\mathbf{x}}_{k,GA}^i)$ (according to Equation 2.29). The N weighted parents could then be employed to create new offspring at the same time step k. That is, enhancing state estimation performance by reshaping the posterior PDF is the main rationale for employing offspring creation schemes (i.e., crossover and mutation) in PF algorithms (Kuptametee et al., 2024).

Wang et al. (2020) and Zhou et al. (2021) suggested that offspring particles should be created only when particle degeneracy of the parent-generation swarm is severe or when quantity ESS_k (found via Equation 2.35) is lower than the preset threshold. However, as previously discussed, a high ESS_k value does not mean that the true non-normalized particle weights are also high. Thus, regardless of the value ESS_k, offspring particles must be created to ensure that the particle swarm will not be trapped at local maximum state values.

To save computation time and to not destroy particle diversity, Yin and Zhu (2015) suggested that all N original particles can be instantly employed as parents without implementing any selection scheme. All N parents at each time step k, however, must first be classified as N_{kH} high-weight parents and N_{kL} low-weight parents where $N_{kH} + N_{kL} = N$; N_{kH} is not necessarily equal to N_{kL}.

According to studies by Yin and Zhu (2015), Yin et al. (2016), Yu et al. (2019), Zhang et al. (2021), and Zhou et al. (2021), the N parents were first sorted by their weights in descending order. The weight of the $[ESS_k]$-th particle that was selected from this new set was then employed as the weight threshold where $[ESS_k]$ was the rounded value of ESS_k that was found via Equation 2.35. Recall that a low ESS_k value denotes severe particle degeneracy as previously shown in Figure 2.2 and the number of high-weight parents N_{kH} must then be low.

Zhou et al. (2021) also employed arithmetic crossover to modify the set of N_{kH} high-weight parents before any offspring particle was found. That is, two new high-

weight parents were created from each of $\lfloor N_{kH}/2 \rfloor$ randomly created pairs of original high-weight parents where $\lfloor \cdot \rfloor$ is the floor symbol. Suppose that high-weight parent $\mathbf{x}_{kH}^{a}$ is paired with another high-weight parent $\mathbf{x}_{kH}^{b}$. Parameter p_c for this pair could be modified and found as:

$$p_c = \begin{cases} p_{c,max}, & \max\left(w_{kH}^{a}, w_{kH}^{b}\right) < \text{Avg}(w_{kH}) \\ p_{c,min} + \dfrac{p_{c,max} - p_{c,min}}{1+\exp\left\{\varepsilon\left[\dfrac{2[\max(w_{kH}^{a}, w_{kH}^{b}) - \text{Avg}(w_{kH})]}{\max(w_{kH}) - \text{Avg}(w_{kH})} - 1\right]\right\}}, & \max\left(w_{kH}^{a}, w_{kH}^{b}\right) \geq \text{Avg}(w_{kH}), \end{cases} \tag{2.46}$$

where $p_{c,max}$ and $p_{c,min}$ denote the upper and lower bounds of parameter p_c, respectively. Parameter ε adjusts the shape of the employed sigmoid function and it must be set carefully by the user. $\text{Avg}(w_{kH})$ and $\max(w_{kH})$ denote the average and the maximum values of the weights of the N_{kH} original high-weight parents, respectively (Zhou et al., 2021). Parameter α for arithmetic crossover in Equation 2.42 was also modified as:

$$\alpha = \max\left(w_{kH}^{a}, w_{kH}^{b}\right) / (w_{kH}^{a} + w_{kH}^{b}), \tag{2.47}$$

where α depends on weights of the two original parents $\mathbf{x}_{kH}^{a}$ and $\mathbf{x}_{kH}^{b}$ (Zhou et al., 2021). Furthermore, Zhou et al. (2021) modified the Metropolis-Hasting (M-H) method (proposed by Hastings (1970)) to find the probability of acceptance for the two new high-weight parents as:

$$\text{POA}\left(\mathbf{x}_{kH,new}^{a}, \mathbf{x}_{kH}^{a}\right) = \min\left(1, \frac{w_{kH.new}^{a}}{\max\left(w_{kH}^{a}, w_{kH}^{b}\right)}\right) \tag{2.48a}$$

$$\text{POA}\left(\mathbf{x}_{kH,new}^{b}, \mathbf{x}_{kH}^{b}\right) = \min\left(1, \frac{w_{kH.new}^{b}}{\max\left(w_{kH}^{a}, w_{kH}^{b}\right)}\right), \tag{2.48b}$$

where, for example, $\mathbf{x}_{kH,new}^{a}$ will always replace $\mathbf{x}_{kH}^{a}$ if its weight is greater than the weights of both original high-weight parents. Otherwise, the probability that $\mathbf{x}_{kH,new}^{a}$ will be accepted can be low if its weight is much lower than $\max\left(w_{kH}^{a}, w_{kH}^{b}\right)$. Note that the weights of the two new high-weight parents and those of the two original high-weight parents must be evaluated according to the same observation $\mathbf{y}_k$. After

arithmetic crossover was employed, all of $2 \times \lfloor N_{kH}/2 \rfloor$ new particles (and the one unpaired original high-weight parent in case quantity N_{kH} is odd) were then gathered together as a set of new high-weight parents (Zhou et al., 2021). However, the weights of some newly found high-weight parents could be lower than the previously found weight threshold and these new inferior particles did not qualify to be employed as high-weight parents.

To create offspring particles, Yin and Zhu (2015) suggested that each low-weight parent must first be paired with a randomly selected high-weight parent; there would be in total N_{kL} pairs of parents. The reason was to prevent pairs of any two parent vectors that had the same state values or equal weights. To follow the principle of GA, Zhou et al. (2021) created a CDF of normalized weights of N_{kH} parent particles and employed the RWS to randomly select a high-weight parent for each low-weight parent. That is, the best high-weight parent had the greatest chance to be selected. However, computation time must be considered. Yin and Zhu (2015), Yin et al. (2016), Yu et al. (2019), Zhang et al. (2021), and Zhou et al. (2021) forced each pair to create a new offspring particle via flat crossover to replace its low-weight parent while its high-weight parent was kept unchanged. That is, parameter p_c in flat crossover was neglected (or set as $p_c = 1$) because the likelihood of finding new high-weight state vectors must be maximized.

Studies by Yin et al. (2016) and Zhou et al. (2021) adaptively adjusted the range of the tuning parameter α for flat crossover in Equation 2.43 as $\alpha \sim U(ESS_k/N, 1)$. If particle degeneracy is severe (or the ESS_k value is low), the bound of α would be large in order to maximize the diversity of offspring particles. On the contrary, if the ESS_k value is high, the state values of each offspring particle would be close to those of its high-weight parent. However, Zhang et al. (2021) used $\alpha \sim U(1 - (ESS_k/N), 1)$ to ensure that state values of each offspring particle will be located around state values of its high-weight parent, especially in case the ESS_k value is low. On the contrary, if the ESS_k value is high, offspring particles could be more freely found within a larger bound for parameter α. A side effect of employing an adaptive bound for parameter α is that new high-weight state vectors may not be searched thoroughly because the search scope sometimes can be very narrow. Consequently, the new population still has chances to be trapped at the local maximum state values of a few high-weight parents.

In traditional GAs, crossover is followed by mutation. Zhou et al. (2021) suggested that all new offspring particles (found via flat crossover) and all high-weight parents (found via arithmetic crossover) should be gathered as a set of N new offspring particles and should enter the mutation process in order to ensure particle diversity. Parameter p_m (which was defined by Katoch et al. (2021) as the probability that the whole offspring vector will mutate), however, was modified to be adaptively calculated for each i-th new offspring particle $\mathbf{x}_{k,off}^{i}$ as:

$$p_m = \begin{cases} p_{m,max}, & w_{k,off}^{i} < \text{Avg}(w_{k,off}) \\ p_{m,min} + \dfrac{p_{m,max} - p_{m,min}}{1+\exp\left\{\varepsilon\left[\dfrac{2[w_{k,off}^{i} - \text{Avg}(w_{k,off})]}{\max(w_{k,off}) - \text{Avg}(w_{k,off})} - 1\right]\right\}}, & w_{k,off}^{a} \geq \text{Avg}(w_{k,off}), \end{cases} \tag{2.49}$$

where $p_{m,max}$ and $p_{m,min}$ denote the upper and lower bounds of parameter p_m, respectively. Parameter ε adjusts the shape of the employed sigmoid function and it must be set carefully by the user. $\text{Avg}(w_{k,off})$ and $\max(w_{k,off})$ denote the average and the maximum values of the weights of the N new offspring particles, respectively (Zhou et al., 2021). Zhou et al. (2021) also suggested that a mutated replica of each offspring should not be immediately accepted. Thus, the M-H method was also applied to find the probability of acceptance of the new mutated offspring as:

$$\text{POA}(\mathbf{x}_{k,off\_m}^{i}, \mathbf{x}_{k,off}^{i}) = \min\left(1, \frac{w_{k,off\_m}^{i}}{w_{k,off}^{i}}\right), \tag{2.50}$$

where $\mathbf{x}_{k,off}^{i}$ is the original offspring particle that is found via flat crossover (Zhou et al, 2021). Note that the weight of offspring $\mathbf{x}_{k,off}^{i}$ and that of its mutated replica $\mathbf{x}_{k,off\_m}^{i}$ must be evaluated according to the same observation $\mathbf{y}_k$.

To save computational cost and to ensure diversity, Park et al. (2009), Wang et al. (2020), and Zhang et al. (2021) suggested that some offspring particles could be found only via crossover while the rest of offspring particles could be found only via mutation. That is, offspring particles that were found by employing only crossover and those found by employing only mutation should coexist within the new-generation

swarm. If every offspring was found only via crossover, the bounds of the swarm could get narrower in the next generation, while undiscovered high-weight state values might be actually located outside the swarm. If only mutation was employed to find every offspring, the algorithm could become inefficient because every new offspring would be found blindly (Kuptametee et al., 2024).

Park et al. (2009) and Wang et al. (2020) manually set the number of crossover-based offspring particles N_{off_c} and the number of mutation-based offspring particles N_{off_m}. These studies, however, had the constraint $N_{off\_c}+N_{off\_m} \leq N$ because parent particles in these studies were not classified as high-weight parents and low-weight parents. This constraint cannot be applied with algorithms that first classify parents by their weights.

Zhang et al. (2021) suggested that the number of offspring particles in each of these two types should depend on the ESS_k value of the parent-generation swarm, while the total number of offspring particles from both types must be equal to the number of low-weight parents N_{kL} where $N_{kL} < N$. When quantity ESS_k is high, there should be no particle whose weight is significantly higher than the weights of the other particles. In this case, flat crossover has greater probability of being selected and employed. On the other hand, when quantity ESS_k is low, the expected number of mutation-based offspring particles should be high. Because the ESS_k value of the parent-generation swarm can be different at each time step, fixed quantities N_{off_C} and N_{off_M} cannot be employed. Thus, Zhang et al. (2021) set the probability that flat crossover would be chosen to ESS_k/N, while probability that the Gaussian mutation would be chosen was set to $1-(ESS_k/N)$. Note that work by Zhang et al. (2021) neglected parameters p_c and p_m in order to maximize the likelihood of finding new high-weight state vectors. That is, for each pair of two parents, if crossover is chosen, an offspring must always be created with $p_c = 1$. In the same way, if mutation is chosen for that pair of parents, an offspring must always be created with $p_m = 1$.

Recall that Zhang et al. (2021) employed mutation to find a new state vector to directly replace the low-weight parent rather than the offspring found by employing flat crossover. Suppose that a high-weight parent $\mathbf{x}_k^{high}$ is paired with a low-weight parent $\mathbf{x}_k^{low}$ and mutation is randomly chosen for the pair. Gaussian mutation (as shown in

Equation 2.44) was then modified by setting the mean values of the Gaussian PDF as the state values of the high-weight parent. That is,

$$\mathbf{x}_k^{off\_m} \sim N\left(\mathbf{x}_k^{high}, \mathbf{\Sigma}\right), \tag{2.51}$$

where the covariance matrix $\mathbf{\Sigma}$ must be carefully designed in order to prevent out-of-bound state values. This ensured that state values of the offspring vector would be located around those of its high-weight parent, while the weight of this new state vector was also expected to have a high value (Zhang et al. 2021).

After we obtain the new population that consists of new N_{kL} offspring particles (that replace their respective low-weight parent) and original N_{kH} high-weight parents, Yin and Zhu (2015) suggested that resampling (or selection) should be employed in order to eliminate particles that are re-considered as having low weights (or low selection probabilities). That is, former high-weight parents whose weights are re-considered as low and new low-weight offspring particles must be eliminated.

Yin et al. (2016) modified work of Yin and Zhu (2015) by removing the resampling step after employing crossover and mutation to obtain a new set of particles. Also, the weight of every particle in this new set was simply reset to be $1/N$ before advancing to the next time step via the state evolution function. Although particle diversity could be preserved and computational cost could be reduced, low-weight particles could still exist and state estimation performance might not be acceptable.

In practice, we should validate the weight of every new offspring particle in order to ensure good state estimation performance. For each pair of two parents, if the weight of an offspring particle is higher than that of its low-weight parent, this offspring will definitely replace its low-weight parent. Otherwise, this offspring should not qualify to exist and state values of its low-weight parent are kept unchanged (Kuptametee et al., 2024; Michalewicz, 1996). Zhang et al, (2021) suggested that an inferior offspring particle (i.e., an offspring particle whose weight is lower than the weight of its low-weight parent) should have a certain likelihood to be accepted and to replace its low-weight parent in order to ensure particle diversity. The probability of acceptance of an offspring particle can be found by applying the M-H method:

$$\mathrm{POA}\left(\mathbf{x}_k^{off}, \mathbf{x}_k^{low}\right) = \min\left(1, \frac{w_k^{off}}{w_k^{low}}\right), \tag{2.52}$$

where $\mathbf{x}_k^{off}$ is the offspring that can be found by employing either only crossover or only mutation (Zhang et al., 2021). If there are too many accepted inferior offspring particles, the state estimation performance can be negatively impacted. Ahwiadi and Wang (2020) and Kuptametee and Aunsri (2022b) suggested that, for each pair of parents, we can retry finding a new offspring particle $\mathbf{x}_k^{off}$ until we obtain the one whose weight is higher than the weight of its low-weight parent. Increasing computational time and cost, however, must be considered.

According to Figure 2.2, severe particle degeneracy can also mean a shortage of high-weight parents. Each low-weight parent then has only a few cases of high-weight parents to pair with. Consequently, the particle swarm can still be trapped at the local maximum state values. Each new offspring particle stays unused after being computed until the last offspring particle is calculated from of the last low-weight parent and its paired high-weight parent. Finally, we can gather all N_{kL} offspring particles (with new evaluated weights) and all unchanged N_{kH} high-weight parents as the new swarm of N particles to estimate the hidden state. That is, we lose the opportunity to employ high-weight offspring particles as new high-weight parents to further promote diversity of state values in the new generation of the particle swarm.

CHAPTER 3

PROPOSED METHOD

In this dissertation, we modify a GA to ensure particle diversity and the likelihood of discovering new high-weight state vectors. The modified GA is also designed to be adaptive to the original parent generation of particle swarm (i.e., the set of N original weighted particles) instead of requiring too many parameters that must be carefully preset. We then integrate the proposed GA into a generic PF algorithm to improve state estimation performance. The following sections in this chapter present the steps of the proposed method.

3.1 Parent Classification

At the beginning of each time step k, each particle $\mathbf{x}_k^i$ is first drawn from the state evolution PDF $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$; its weight w_k^i is found as $p(\mathbf{y}_k|\mathbf{x}_k^i)$ (as per Equation 2.29). As suggested by Yin and Zhu (2015), all N weighted particles should participate in offspring creation without employing a selection process to ensure particle diversity. All particles, however, must first be classified as high-weight or low-weight parents. The benefits of particle classification are: (1) to prevent having pairs of any two identical parents and (2) to prevent high-weight parents from being replaced by new offspring particles. The proposed method computes the weight threshold for classifying parents at time step k as:

$$w_k^{thr} = \frac{1}{N}\sum_{i=1}^{N} w_k^i, \tag{3.1}$$

which is the average of the true non-normalized weights of all N particles in the original parent generation at time step k. Particles whose weights are not lower than w_k^{thr} will be classified as high-weight parents, while the others are classified as low-weight parents. Equation 3.1 does not require weight sorting and computation time can then be saved. Furthermore, the weight threshold found in work by Yin and Zhu (2015), Yin et

al. (2016), Yu et al. (2019), Zhang et al. (2021), and Zhou et al. (2021) as the weight of the $[ESS_k]$-th best particle is not always adequately high. Suppose that there are 10 particles whose weights are sorted in a descending order as: 0.28, 0.19, 0.14, 0.12, 0.08, 0.06, 0.05, 0.04, 0.03 and 0.01. The ESS value of this swarm calculated according to Equation 2.35 will be approximately 6.1125 and $[6.1125] = 6$. Thus, the weight of the sixth best particle, 0.06, is then employed as weight threshold, while the value 0.06 is lower than the average of all ten weights, 0.1.

Note that the number of low-weight parents and the number of high-weight parents can be different and time-varying. We denote the number of low-weight parents at time step k and the number of high-weight parents at time step k as variables N_{kL} and N_{kH}, respectively. The pseudocode for parent classification is also provided in Figure 3.1.

Input: N particles ($\mathbf{x}_k$) and their true weights (w_k)
Output: N_{kL} low-weight parents ($\mathbf{x}_{kL}$) and N_{kH} high-weight parents ($\mathbf{x}_{kH}$) where
$N_{kL} + N_{kH} = N$
$w_k^{thr} \leftarrow \sum_{i=1}^{N} w_k^i$ %Employ average of all true weights as the threshold
$l \leftarrow 0$ %Initialize the index number of low-weight parents
$h \leftarrow 0$ %Initialize the index number of high-weight parents
for $i \in \{1, \dots, N\}$ **do**
    if $w_k^i \geq w_k^{thr}$ **then** %Classify the particle as a high-weight parent
        $h \leftarrow h + 1$
        $\mathbf{x}_{kH}^h \leftarrow \mathbf{x}_{k,old}^i$
        $w_{kH}^h \leftarrow w_{k,old}^i$
    else %Classify the particle as a low-weight parent
        $l \leftarrow l + 1$
        $\mathbf{x}_{kL}^l \leftarrow \mathbf{x}_k^i$
        $w_{kL}^l \leftarrow w_k^i$
    end if
end for

Figure 3.1 A pseudocode for parent classification

3.2 Parent Pairing

Each pair of parents is created sequentially. Each low-weight parent randomly selects a high-weight parent to pair with. We allow each high-weight parent to be selected multiple times. Otherwise, in case particle degeneracy occurs when the number of high-weight parents is smaller than that of low-weight parents as shown in Figure 2.2, some low-weight parents will not have any high-weight parents to pair with.

According to the traditional principle of GAs, the fittest individual has the highest chances to survive and to produce new offspring individuals (Katoch et al., 2021; Larose,2006; Michalewicz, 1996). Thus, the parent particle with the maximum weight is supposed to be selected most often or to pair with low-weight parents in greatest numbers (Zhou et al., 2021). However, the state values of the maximum-weight particle that we have on hand are not necessarily located around the global maximum state values (that are located at the highest peak of the true unknown posterior PDF). That is, the true weight of the maximum-weight particle that we have on hand may be actually low (Kuptametee et al., 2024).

Suppose that we need to create the a-th pair of parents where $a \in \{1, \dots, N_{kL}\}$. We draw an index number $b \sim U\{1, \dots, N_{kH}\}$ to select the b-th high-weight parent $\mathbf{x}_{kH}^{b}$ to pair with the a-th low-weight parent $\mathbf{x}_{kL}^{a}$. That is, we set the selection probability of each high-weight parent to be the same because we need to mitigate the chances that the new offspring particles will be trapped around the state values of the maximum-weight particle (or local maximum state values). Note that each index number b does not denote ranking orders of the high-weight parents because weight sorting was not employed during particle classification.

3.3 Offspring Creation

Zhang et al. (2021) suggested that there are two types of offspring particles that should be computed together in order to obtain the new generation of particle swarm: (1) offspring particles that are calculated using only flat crossover (proposed by Radcliffe (1990) as in Equation 2.43), and (2) offspring particles that are computed

using only modified Gaussian mutation (proposed by Zhang et al. (2021) as in Equation 2.51). This scheme ensures that new high-weight state vectors will be identified in the search space. However, the number of offspring particles from each type should depend on the ESS_k value of the original parent-generation particle swarm. When the ESS_k value is high, the variance of all particle weights is supposed to be low and flat crossover should be preferred to find new offspring particles whose state values are located within the bounds of the swarm. On the other hand, when the ESS_k value is low (or particle degeneracy is severe), there are only few particles whose weights are significantly higher than the weights of the other particles, as previously shown in Figure 2.2. Modified Gaussian mutation should be preferred to find new offspring particles that are located around the state values of these few high-weight parents. Thus, we adopt the Gaussian mutation modified by Zhang et al. (2021) by setting the probability of choosing flat crossover for each pair at time step k as:

$$\gamma_k = \frac{\left(\sum_{i=1}^{N} w_k^i\right)^2}{N \times \sum_{i=1}^{N} \left(w_k^i\right)^2}, \tag{3.2}$$

while the probability of choosing modified Gaussian mutation for each pair at time step k is $1 - \gamma_k$. That is, the expected number of crossover-based offspring particles to be found at time step k is:

$$N_{kC,exp} = N_{kL} \times \gamma_k, \tag{3.3}$$

and the expected number of mutation-based offspring particles to be found at time step k is:

$$N_{kM,exp} = N_{kL} \times (1 - \gamma_k). \tag{3.4}$$

All high-weight parents are always kept unchanged, while each low-weight parent is supposed to be replaced with its offspring in case the latter has a higher weight. However, the new state vector may not always have a higher weight. We must first find the offspring candidate particle $\mathbf{x}_{cand}$ from a pair of the low-weight parent $\mathbf{x}_{kL}^{a}$ and the high-weight parent $\mathbf{x}_{kH}^{b}$.

Let a random number $u \sim U(0,1)$ be drawn for selecting a GA operator for offspring creation. If $u \leq \gamma_k$, flat crossover will be chosen and the offspring candidate can be calculated as:

$$\mathbf{x}_{cand} = \alpha \mathbf{x}_{kH}^{b} + (1-\alpha)\mathbf{x}_{kL}^{a}, \tag{3.5}$$

where variable $\alpha \sim U(0,1)$ is the state value tuning parameter. If α is low, the state values of the candidate $\mathbf{x}_{cand}$ will be close to those of the low-weight parent $\mathbf{x}_{kL}^{a}$. In contrast, the state values of the candidate $\mathbf{x}_{cand}$ will be close to those of the high-weight parent $\mathbf{x}_{kH}^{b}$ if α is high. Equation 3.5 can be employed only when the two parent vectors have the same size. Equation 3.5 can then be generalized for finding each m-th new state value as:

$$x_{cand,m} = \alpha x_{kH,m}^{b} + (1-\alpha)x_{kL,m}^{a}, \tag{3.6}$$

where tuning value α must be the same for every m-th vector component, where $m \in \{1, \ldots, \min(d_{kL}^{a}, d_{kH}^{b})\}$. Quantities d_{kL}^{a} and d_{kH}^{b} denote the size of the low-weight parent $\mathbf{x}_{kL}^{a}$ and the size of the high-weight parent $\mathbf{x}_{kH}^{b}$, respectively. That is, only the first $\min(d_{kL}^{a}, d_{kH}^{b})$ vector components of the two parents can be paired. Note that the two parent state values $x_{kL,m}^{a}$ and $x_{kH,m}^{b}$ must have same data type or unit, while the whole vector may consist of state values with different data types or units.

In case $d_{kH}^{b} > d_{kL}^{a}$, the $[\min(d_{kL}^{a}, d_{kH}^{b}) + 1]$-th through the d_{kH}^{b}-th components of the high-weight parent $\mathbf{x}_{kH}^{b}$ will be left unused. In case $d_{kH}^{b} < d_{kL}^{a}$, every vector component of the high-weight parent $\mathbf{x}_{kH}^{b}$ will be employed in Equation 3.6. Then, the $[\min(d_{kL}^{a}, d_{kH}^{b}) + 1]$-th through the d_{kL}^{a}-th components of the low-weight parent $\mathbf{x}_{kL}^{a}$ will be concatenated to the newly found candidate $\mathbf{x}_{cand}$ without being changed. That is, the size of the candidate $\mathbf{x}_{cand}$ that is found using flat crossover must be the same as that of the low-weight parent $\mathbf{x}_{kL}^{a}$. Note that, although the two parent vectors have different sizes in this case, all of their respective state values must have the same data type or unit.

Recall that the random value $u \sim U(0,1)$ is drawn to select a GA operator for offspring creation. If $u > \gamma_k$, modified Gaussian mutation will be chosen and the offspring candidate can be found as:

$$\mathbf{x}_{cand} \sim N(\mathbf{x}_{kH}^{b}, \boldsymbol{\Sigma}), \tag{3.7}$$

where $\boldsymbol{\Sigma}$ is a $d_{kH}^{b} \times d_{kH}^{b}$ covariance matrix that can be designed by the user. Recall that, in this mutation scheme, the state values of the high-weight parent $\mathbf{x}_{kH}^{b}$ are employed as mean values of the Gaussian PDF. Thus, the size of the candidate $\mathbf{x}_{cand}$ that is found using modified Gaussian mutation must be same to that of the high-weight parent $\mathbf{x}_{kH}^{b}$.

Finally, we find the a-th offspring particle that will replace its low weight parent $\mathbf{x}_{kL}^{a}$ as:

$$\mathbf{x}_{k}^{a,off} = \begin{cases} \mathbf{x}_{cand}, & p(\mathbf{y}_k|\mathbf{x}_{cand}) > p(\mathbf{y}_k|\mathbf{x}_{kL}^{a}) \\ \mathbf{x}_{kL}^{a}, & p(\mathbf{y}_k|\mathbf{x}_{cand}) \leq p(\mathbf{y}_k|\mathbf{x}_{kL}^{a}), \end{cases} \tag{3.8}$$

where state values of the candidate $\mathbf{x}_{cand}$ will be accepted and assigned to the offspring particle only if the weight of the candidate $\mathbf{x}_{cand}$ is higher than the weight of the low-weight parent $\mathbf{x}_{kL}^{a}$. Otherwise, the low-weight particle $\mathbf{x}_{kL}^{a}$ assigns its state values to the offspring particle without being changed.

3.4 Evolution of High-weight Offspring Particles

Initially, particles are classified according to the threshold w_k^{thr} that is calculated using Equation 3.1. Consequently, there are two disjoint sets of particles: high-weight particles (red-line circles) and low-weight particles (blue-line circles) as shown in Figure 3.2. When offspring particles are found, a set of offspring particles (yellow-line circle) must be created. Some offspring particles are actually new state vectors while the others are just replicas of their respective low-weight parents. That is, the set of replicas of low-weight particles is a subset of the set of offspring particles. Also, some offspring particles may have weights that are not smaller than the threshold w_k^{thr}. That is, these offspring particles satisfy the condition of being classified as high-weight parents and they can then be employed along with original members of the set

of high-weight parents. Thus, the set of high-weight particles and the set of offspring particles are not disjoint. When all N_{kL} offspring particles are found, the original set of low-weight particles becomes an empty set and can be removed. Recall that the weight threshold w_k^{thr} must stay fixed once it is calculated via Equation 3.1.

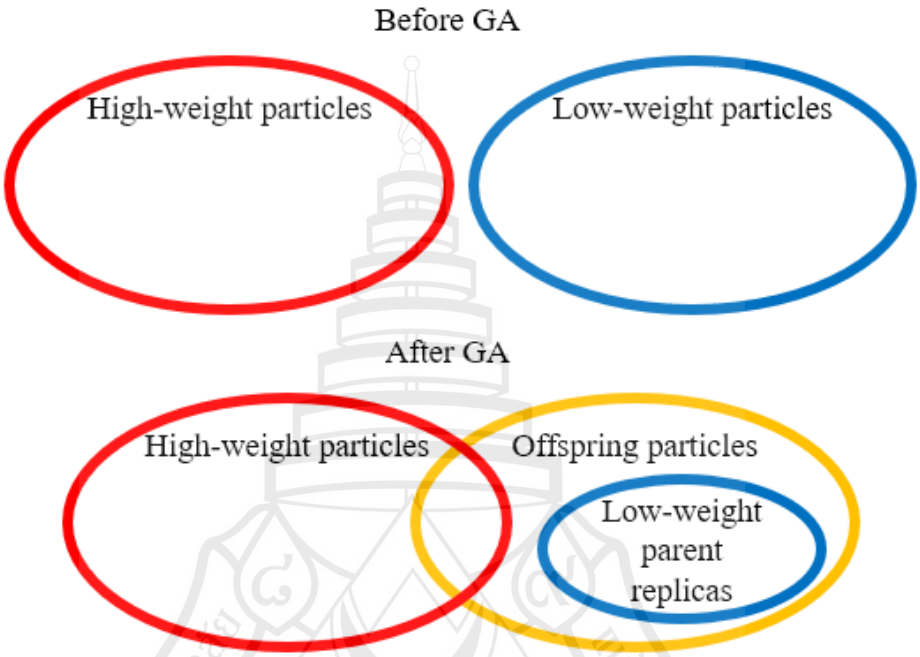


Figure 3.2 Euler diagrams of sets of particles before and after employing GA

As previously discussed, when particle degeneracy is severe, each low-weight parent has a few choices of high-weight parents to randomly choose and pair with. Normally, each offspring particle can be obtained sequentially from each pair and the computed offspring particles are stored unused until we obtain all N_{kL} offspring particles. Some of the stored offspring particles may have high weights and we should make use of them to ensure diversity of the rest of the offspring particles. In other words, the low-weight parents should be offered more choices of high-weight parents to pair with.

Figure 3.3 demonstrates the process of finding the first offspring particle (i.e., particle $\mathbf{x}_k^{a,off}$ with index number $a = 1$) where the digit on each particle denotes its index number. The selected low-weight parent also generates a replica that is employed during weight comparison against its offspring. If the weight of the offspring is smaller than or equal to the weight of its low-weight parent, this offspring will be rejected and

the state values of its low-weight parents will replace the new state values. Then, this offspring is regarded as the replica of its low-weight parent.

If the weight of the offspring is higher than the weight of its low-weight parent, this offspring will be accepted. Also, if the weight of the offspring is not smaller than the preset threshold w_k^{thr} (computed via Equation 3.1), it will evolve to a new high-weight parent with an additional probability to be selected and paired with the rest of the low-weight parents. Otherwise, the offspring will be stored unused because its weight is not high enough for it to be promoted as a new high-weight parent.

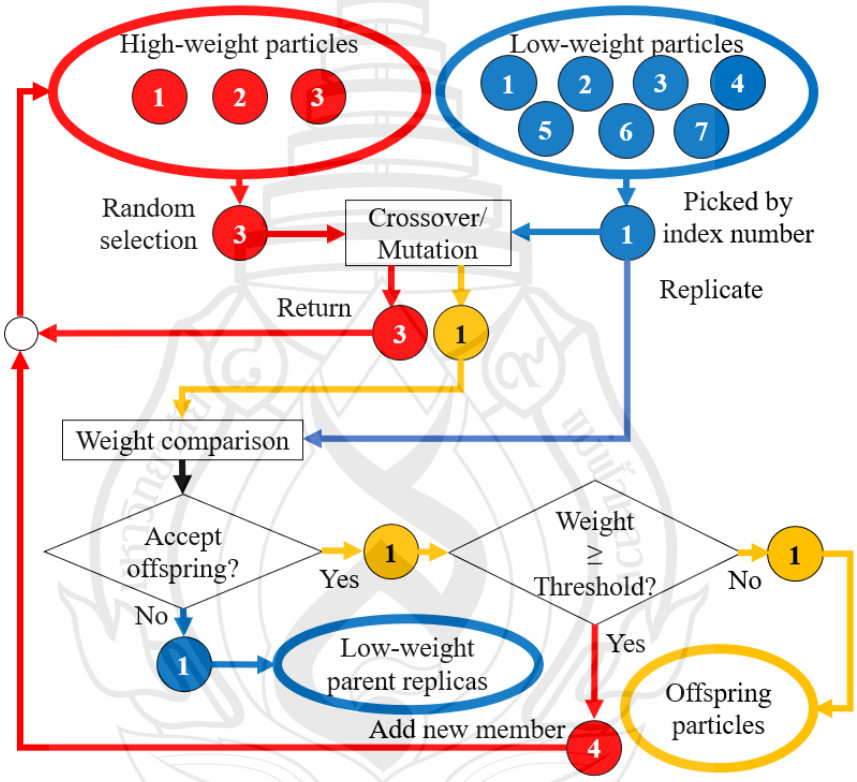


Figure 3.3 The process of finding an offspring particle in the proposed method

We iterate the overall process in Figure 3.3 until there are no original low-weight parents (not their replicas) left. Finally, we gather all N weighted particles from all three sets (high-weight particles, offspring particles, and low-weight parent replicas) as the new-generation population (or particle swarm) at time step k and we employ this new population to infer (or estimate) the true state at time step k. The pseudocode for improving the low-weight parents (i.e., offspring creation) in the proposed method is also shown in Figure 3.4.

Input: N_{kL} low-weight parents ($\mathbf{x}_{kL}$) and N_{kH} high-weight parents ($\mathbf{x}_{kH}$)

Output: N_{kL} offspring particles and N_{kH} original high-weight parents ($\mathbf{x}_{kH}$)

$\gamma_k \leftarrow \left(\sum_{i=1}^{N} w_k^i\right)^2 / \left[N \times \sum_{i=1}^{N} \left(w_k^i\right)^2\right]$ %Probability of choosing flat crossover

$h \leftarrow N_{kH}$ %Initial number of high-weight parents

for $a \in \{1, \dots, N_{kL}\}$ **do**

  $b \sim U\{1, \dots, h\}$

  $u \sim U(0,1)$

  if $u \leq \gamma_k$ **then** %Employ flat crossover

    $\alpha \sim U(0,1)$

    $d \leftarrow \min(d_{xL}, d_{xH})$ %Number of pairable vector components

    for $m \in \{1, \dots, d\}$ **do**

      $x_{cand,m}^a \leftarrow \alpha x_{kH,m}^b + (1-\alpha) x_{kL,m}^a$

    end for

    if $d_{xL} > d$ %In case low-weight parent is longer

      for $m \in \{d+1, \dots, d_{xL}\}$ **do**

      $x_{cand,m}^a \leftarrow x_{kL,m}^a$ %Inherit unpaired vector components

      end for

    end if

  else %Employ mutation

    $\mathbf{x}_{cand}^a \sim N(\mathbf{x}_{kH,m}^b, \boldsymbol{\Sigma}_{GM})$

  end if

  $w_{cand}^a \leftarrow p(\mathbf{y}_k | \mathbf{x}_{cand}^a)$ %Weight of the a-th offspring candidate

  if $w_{cand} \geq w_k^{thr}$ **then** %Add new high-weight parent

    $h \leftarrow h + 1$

    $\mathbf{x}_{kH}^h \leftarrow \mathbf{x}_{cand}^a$

    $w_{kH}^h \leftarrow w_{cand}^a$

  else if $w_{cand}^a \geq w_{kL}^a$ **then** %Replace the low-weight parent

    $\mathbf{x}_{kL}^a \leftarrow \mathbf{x}_{cand}^a$

    $w_{kL}^a \leftarrow w_{cand}^a$

  else

    Keep the low-weight parent $\mathbf{x}_{kL}^a$ and its weight unchanged

  end if

end for

Figure 3.4 A pseudocode for offspring creation

Suppose that the number of the original high-weight parents at time step k is $N_{kH,old}$ and the number of the new high-weight offspring particles at time step k whose weights are not lower than the threshold as computed by Equation 3.1 is $N_{kH,new}$ (which is initially zero). In theory, the computational complexity of adding a new high-weight offspring into an array (or a set) of high-weight parents is $O(N_{kH,old} + N_{kH,new})$ because we have to create a new array with a larger size before we move the $N_{kH,old} + N_{kH,new}$ high-weight particles (excluding the newest high-weight offspring at the moment) to the new array (Lewis & Chase, 2014). It is possible that all N_{kL} new candidates will not only be accepted as new offspring but also evolve to be new high-weight parents. Thus, we can create an array with size $N \times d_x$ in advance to store both the original and the new high-weight particles. The complexity of adding (or pushing) a new particle into this "maximum size" array is only $O(1)$ because we do not need to create a new array once each high-weight offspring is obtained (Lewis & Chase, 2014).

CHAPTER 4

SIMULATION RESULTS

This chapter presents results of state estimation of the proposed method in simulation state-space models. There are two experiments conducted where the state-space model is: (1) one-dimensional, and (2) multidimensional. The size of the state vectors in each of these experiments is, however, not time-varying.

The state-of-the-art algorithms selected for experiments in this chapter are as follows.

The traditional particle filter called sequential importance resampling particle filter (SIR-PF) employs stochastic universal sampling (SUS) (i.e., systematic resampling) to eliminate low-weight particles at every time step. This selection scheme is also employed by the auxiliary SIR particle filter (ASIR-PF) proposed by Pitt and Shephard (1999) to resample the auxiliary particles.

The adaptive fission particle filter (AFPF) proposed by Han et al. (2015) sets the minimum number of replicas to be created from each of N particles (excluding themselves) to $N_{rep,min} = 2$. A fission factor is also employed to tune variance values according to the weight of each original particle according to Equation 2.39.

The genetic optimization resampling particle filter (GORPF) proposed by Zhou et al. (2021) creates new offspring only when the effective sample size (ESS) found according to the new weights of N post-roughening parents is lower than $0.7N$. Roughening is employed according to Equation 2.38 with a tuning parameter $\beta = 0.2$ as suggested by Gordon et al. (1993). In parent classification, the number of high-weight parents is $N_{kH} = [ESS_k]$ where $[\cdot]$ is the rounding symbol, e.g., $[3.4] = 3$ and $[3.5] = 4$. Arithmetic crossover is employed to create new high-weight parents with parameters set as: $p_{c,min} = 0.6$, $p_{c,max} = 0.9$ and $\varepsilon = 9.903438$, according to Equation 2.46, while Equations 2.48a and 2.48b are employed to accept or reject the two new high-weight parents. The ESS_k of all N particles must be re-evaluated and flat crossover (in Equation 2.43) can then be employed with $\alpha \sim U(ESS_k/N, 1)$ where each new offspring must replace its low-weight parent, while each high-weight parent is

randomly selected according to a CDF of the normalized weights of N_{kH} high-weight parents. Finally, Gaussian mutation is employed to perturb state values of every particle (including N_{kH} high-weight parents and N_{kL} new offspring particles) according to Equation 2.44 with the parameter p_m calculated via Equation 2.49. Equation 2.49 is employed with the following parameters: $p_{c,min} = 0.6$, $p_{c,max} = 0.9$, and $\varepsilon = 9.903438$. Finally, Equation 2.50 is employed to accept or reject the mutated replica of each particle.

The intelligent particle filter (IPF) proposed by Zhang et al. (2021) treats the $[ESS_k]$ best particles as high-weight parents while the rest are treated as low-weight parents, similar to GORPF. The probability that each of N_{kH} high-weight parents will be selected is set to be uniform, similar to our proposed method. If flat crossover is chosen with probability ESS_k/N, flat crossover will be employed according to Equation 2.43 with $\alpha \sim U(1 - (ESS_k/N), 1)$. Modified Gaussian mutation can be chosen and employed according to Equation 2.51 with probability $1 - (ESS_k/N)$. Equation 2.52 is employed for allowing the new offspring to replace its low-weight parent or not.

4.1 One-dimensional State Estimation

In this section, we choose a benchmark non-linear state-space model from work by Gordon et al. (1993) to perform a one-dimensional (1-D) state estimation experiment. The state evolution function and the observation function of this model are:

$$x_k = f_{k-1}(x_{k-1}, u_{k-1}) = 0.5x_{k-1} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 8\cos[1.2(k-1)] + u_{k-1}, \quad (4.1)$$

and

$$y_k = g_k(x_k, v_k) = 0.05x_k^2 + v_k, \quad (4.2)$$

where quantity u_{k-1} is additive state evolution noise that updates old state values at time step $k-1$ and quantity v_k is additive observation noise at time step k, respectively. The weight of each particle x_k^i in this system can be found from the likelihood function:

$$w_k^i = \exp\left(\frac{-(y_k - g_k(x_k^i))^2}{2\sigma_{v_k}^2}\right), \tag{4.3}$$

where $g_k(x_k^i)$ denotes the i-th predicted observation data value and $\sigma_{v_k}^2$ denotes the variance of observation noise v_k. The parameters configured for this 1-D state-space model are provided in Table 4.1

Table 4.1 Parameters for the 1-D state estimation experiment

Symbol	Meaning	Value
x_0	Initial state	0
$p(x_0)$	Initial prior PDF	$N(0, 2)$
u_{k-1}	State evolution noise	Draw from $N(0, 2)$
v_k	Observation noise	Draw from $N(0, 2)$
K	Number of time steps	100
R	Number of simulation runs	50

Some configurations in this experiment were specifically made for selected state-of-the-art algorithms. The AFPF employs a variance σ^2 of 2 for perturbing the state values of each created replica. The GORPF, IPF and the proposed method also employ a variance σ^2 of 2 in their respective mutation schemes.

In this experiment, the state was inferred using the weighted mean (WM) of sample state values. To obtain a fair state estimation comparison using WM, the weight of each new particle in the final set that was obtained after resampling in SIR-PF and ASIR-PF, after roughening in GORPF (only when $ESS_k \geq 0.7N$), after offspring creation in GORPF (only when $ESS_k < 0.7N$), IPF and the proposed method, must be re-evaluated via Equation 2.29 before we estimate the state at the end of any time step k, instead of leaving particle weights as $1/N$.

The results of 1-D state estimation using WM with $N = 100$ particles are shown in Figures 4.1 and 4.2, where the true states (shown as a black curve with black dots) are plotted against estimated tracks of 50 simulation runs (shown as magenta curves). Figure 4.1 shows comparison results obtained from non-GA-based PF algorithms (i.e.,

SIR-PF, ASIR-PF and AFPF), while Figure 4.2 shows comparison results obtained from GA-based PF algorithms (i.e., GORPF, IPF and the proposed PF).

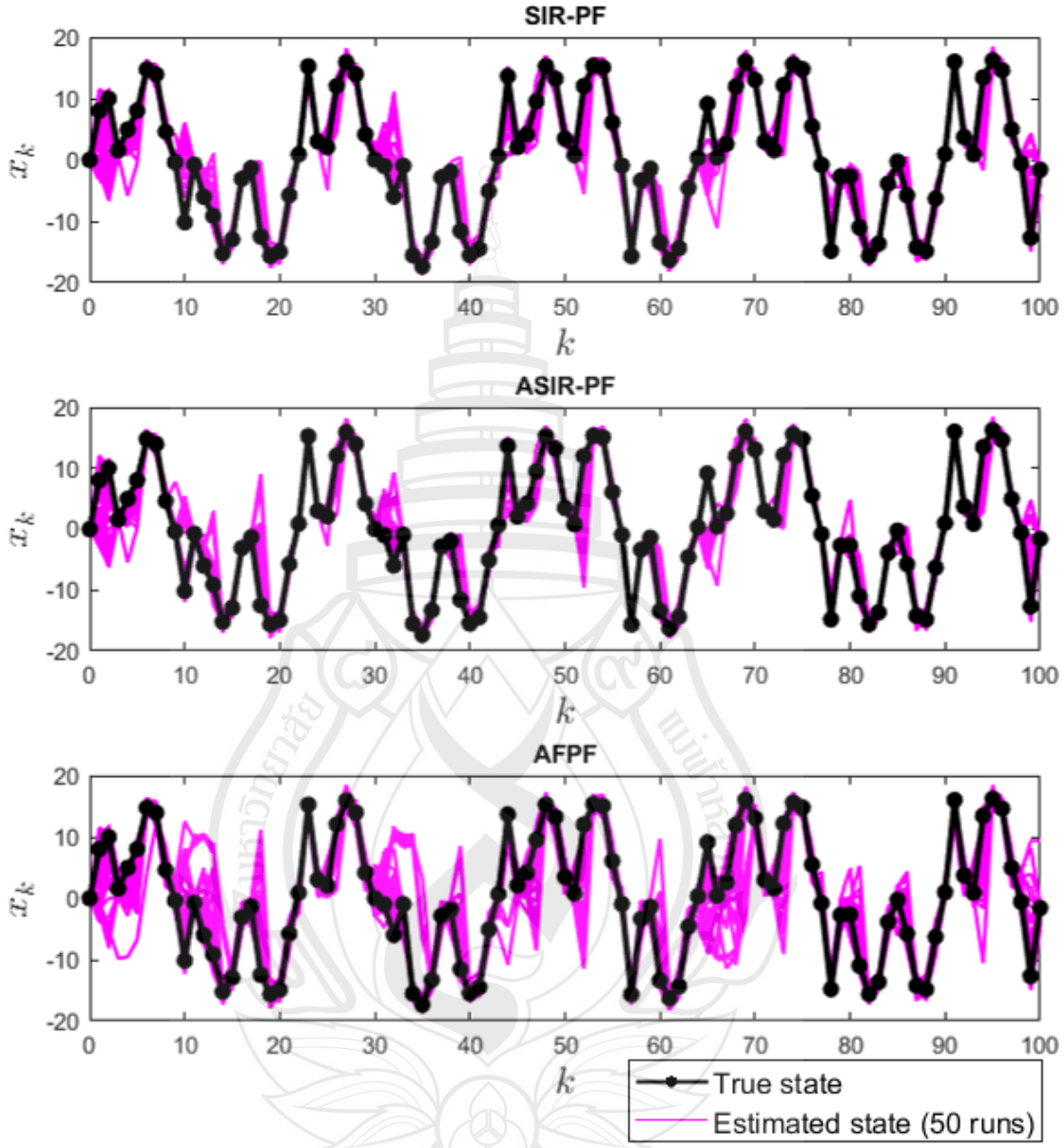


Figure 4.1 Comparison of 1-D state estimations via WM by employing non-GA-based PF algorithms

In Figure 4.1, results obtained from non-GA-based PF algorithms (i.e., SIR-PF, ASIR-PF and AFPF show significant errors: the estimated tracks deviate from the true state track. In ASIR-PF, the estimated tracks are more erroneous than those of the SIR-PF at, for example, around time steps 17-18, time steps 51-52, and time steps 79-80. The reason is that the ASIR-PF further perturbs state values of resampled auxiliary

particles blindly. The AFPF, however, delivered the most erroneous estimation results: many parts of the estimated tracks clearly deviate from the track of the true state. In AFPF, each high-weight particle created replicas at higher numbers, but the state values of these replicas were not perturbed much from the state values of their original copy because the fission factor tuned the variance of the Gaussian PDF that was employed for drawing random perturbing values to be low. Each of the low-weight particles, in contrast, created fewer replicas where the fission factor value and the variance of the Gaussian PDF were high. Thus, the particle swarm in AFPF, even after weight sorting and keeping only the N best particles, still could be trapped at the local maximum state values.

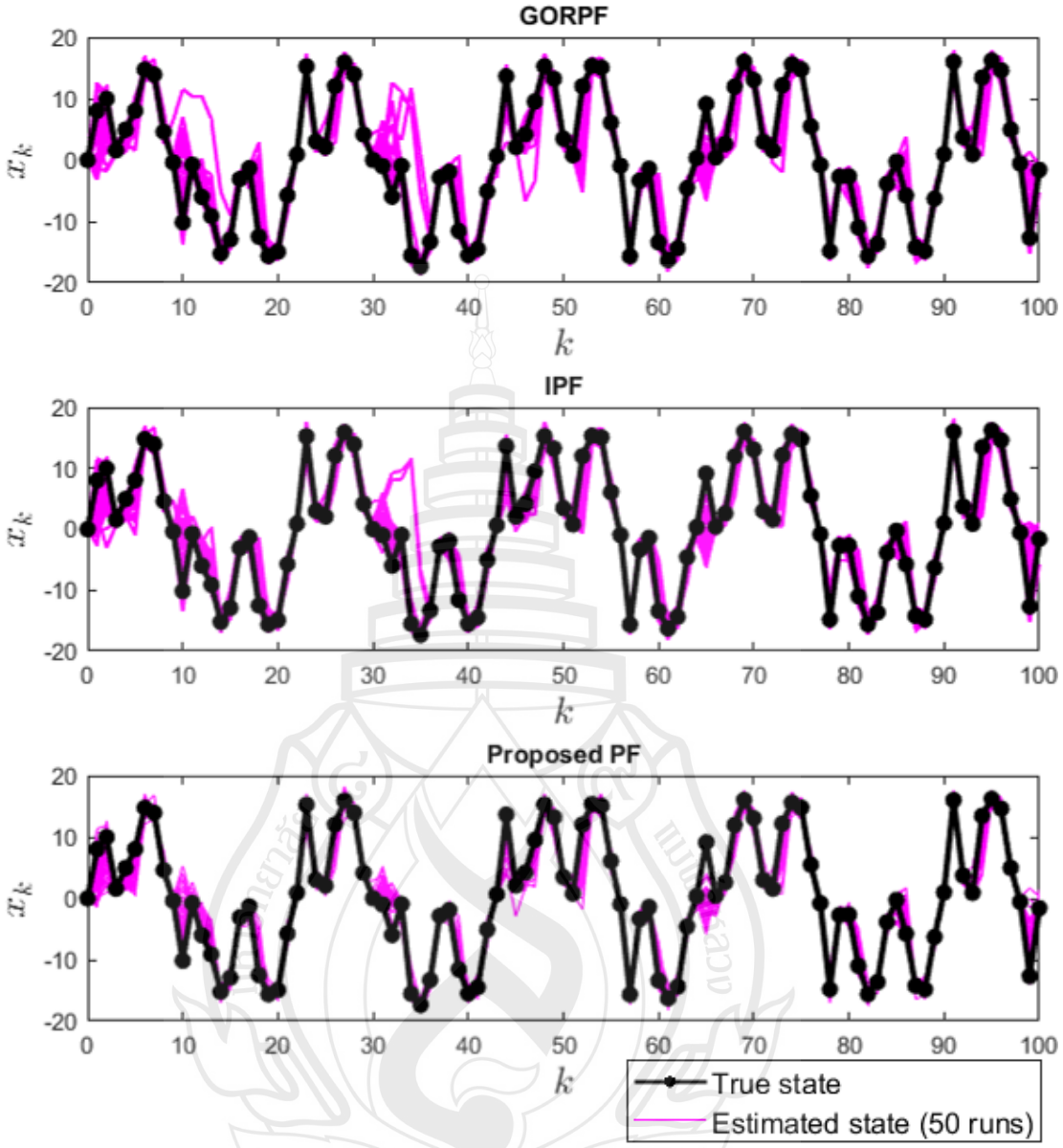
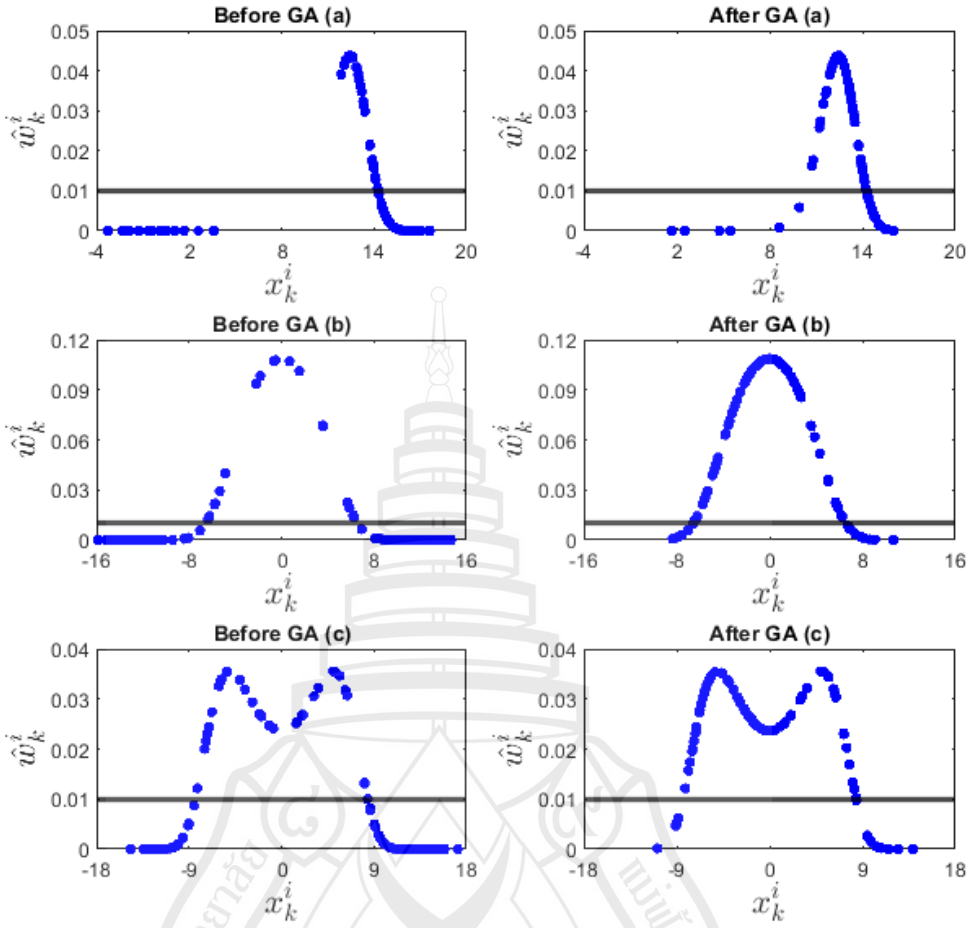


Figure 4.2 Comparison of 1-D state estimations via WM by employing GA-based PF algorithms

Figure 4.2 compares results of 1-D state estimation using WM obtained from GA-based PF algorithms: GORPF, IPF and the new method. The GORPF, at some time steps, employed only roughening to diversify the state values of the selected parents (or resampled particles) because quantity ESS_k was not smaller than the preset threshold. As previously discussed, a high ESS_k does not mean that all (or most of) particles are located at regions of high-weight state values. Thus, the GORPF lost the opportunity to find new and better particles to enhance state estimation at some time steps and,

consequently, delivered erroneous results, as shown in time steps 8-15 and 30-36. The IPF accepted some new inferior low-weight particles to be members of the new generation of swarm. These inferior particles then participated in the state estimation and negatively affected its performance, as shown in time steps 30-36. Thus, the new method is obviously superior to the others because this method always rejected every offspring candidate that was inferior to its respective low-weight parent. Also, potential offspring particles were employed as new high-weight parents instead of being kept unused until we needed to estimate the state.

Figure 4.3 shows the effects of employing the proposed method for posterior PDF reshaping at selected time steps of one run. To keep the values of the y-axis scale (i.e., normalized weights) consistent as before and after employing our method, we normalized the weights of pre-reshaping the posterior PDFs (shown on the left-hand side). The dark gray lines denote the normalized weight threshold at the average (or $1/N$ where $N = 100$). Some low-weight particles can be seen to have been relocated to the new positions where high-weight state values existed because they were replaced by offspring particles that had higher weights. Note that the weights of particles in post-reshaping PDFs (shown on the right-hand side) had not been re-normalized because we need to present clear comparisons between before and after employing the new method.



Note (a) Time step $k = 6$, (b) Time step $k = 31$, (c) Time step $k = 86$

Figure 4.3 Posterior PDFs are reshaped after employing the proposed method

Table 4.2 presents a comparison of the 1-D state estimation performance previously shown in Figures 4.1 and 4.2 in terms of numerical error measurements. Root-mean-squared errors (RMSE) and mean absolute errors (MAE) were employed to assess the state estimation performance. We employ their averages values that can be found as:

$$\text{Avg(RMSE)} = \frac{1}{R}\sum_{r=1}^{R}\sqrt{\frac{1}{K}\sum_{k=1}^{K}|x_k - \hat{x}_{k,r}|^2} \tag{4.4}$$

and

$$\text{Avg(MAE)} = \frac{1}{KR}\sum_{r=1}^{R}\frac{1}{K}\sum_{k=1}^{K}|x_k - \hat{x}_{k,r}|, \tag{4.5}$$

where x_k denotes the true state at time step k from all K time steps, while $\hat{x}_{k,r}$ denotes the estimated state at time step k of the r-th simulation run from all R runs (Aunsri et al., 2021; Kuptametee & Aunsri, 2022b).

Table 4.2 Numerical error measurements in 1-D state estimation

PF	Avg(RMSE)	Var(RMSE)	Avg(MAE)	Var(MAE)
SIR-PF	3.1113	0.1471	1.6290	0.0277
ASIR-PF	3.1922	0.1790	1.5644	0.0348
AFPF	4.0221	1.0266	2.2174	0.2540
GORPF	2.9023	0.2758	1.5145	0.0461
IPF	2.6131	0.1692	1.4481	0.0286
Proposed	**2.4507**	**0.0185**	**1.3998**	**0.0077**

The average numerical errors of non-GA-based PF algorithms were higher than those of GA-based PF algorithms. The average RMSE of the SIR-PF was lower than that of the ASIR-PF but the average MAE of the SIR-PF was higher than that of the ASIR- PF. However, the variances of the numerical errors of the SIR-PF were lower than those of the ASIR-PF for both RMSEs and MAEs. The AFPF yielded the highest averages of errors and the highest variances of errors. This showed that blind state value perturbation was ineffective.

Results obtained from the GORPF were more erroneous than results delivered from the other two GA-based PF algorithms, the IPF and the proposed method. Although the average errors of the GORPF were lower than those of the SIR-PF and ASIR-PF, the variance of RMSEs and MAEs of the GORPF were higher than not only those of the SIR-PF and the ASIR-PF but also those of the IPF and the proposed PF. In roughening which was employed in the GORPF, the state values of N resampled particles were perturbed according to the same set of adaptive variance values, which could be different for each vector component at each time step via Equation 2.37. That is, variance values in roughening could vary for each time step. This showed that the adaptability of the IPF and the proposed PF without pre-setting too many GA parameters provided better performance compared to the GORPF. The average RMSE and MAE obtained by the IPF were lower than those of the SIR-PF, but the variance of

RMSEs and MAEs of the IPF are higher than those of the SIR-PF. The reason was that the IPF sometimes accepted candidate particles as new offspring particles, while their weights were lower than those of their respective low-weight parents. The performance of the proposed method was better and more reliable than the other algorithms as shown by having the lowest averages and variances of errors in this experiment.

The computation time for 1-D state estimation for each PF algorithm where $N = 100$ was also measured and provided in Table 4.3. The average computation time can be found as:

$$\text{Avg(Time)} = \frac{1}{KR}\sum_{r=1}^{R}\sum_{k=1}^{K} t_{k,r}, \tag{4.6}$$

where $t_{k,r}$ denotes the computation time (measured in seconds) at time step k of the r-th simulation run. Quantities K and R denote the total number of time steps and the total number of simulation runs, respectively (Aunsri et al., 2021).

Table 4.3 Computation time in 1-D state estimation

PF	Minimum	Average	Maximum	Variance ($\times 10^{-6}$)
SIR-PF	**0.0013**	**0.0022**	**0.0175**	**2.1685**
ASIR-PF	0.0017	0.0031	0.0190	3.2025
AFPF	0.0019	0.0040	0.0436	11.2781
GORPF	0.0023	0.0048	0.1587	81.8221
IPF	0.0016	0.0038	0.0431	9.4376
Proposed	0.0014	0.0031	0.0420	3.8568

The SIR-PF runtime was the shortest because, after resampling and state estimation, the SIR-PF just moved to next time step by updating state values of N resampled particles via the state evolution function. This differs from perturbing the state values to estimate the state with diversified particles. The ASIR-PF required slightly longer computation time compared to the SIR-PF because the ASIR-PF blindly perturbed the state values of the N resampled auxiliary particles to obtain the N new particles employed in state estimation. In the GORPF resampling (i.e., parent selection) and roughening were always employed at every time step. This scheme is more

complicated than perturbing the state values of N resampled particles with fixed-variance Gaussian random values as implemented in the ASIR-PF. Thus, the minimum computation time spent by the GORPF was the largest. The AFPF, IPF and the proposed PF did not employ any traditional selection (or resampling) scheme. However, the AFPF required substantial time to find new state values of replicas of each of the N original particles. The total number of newly created replicas (with perturbed state values) could exceed N, while weight sorting was also required to keep the best N particles. Also, the AFPF calculated the fission factor value of each of the N original particles which depended on the particle weights as in Equation 2.39.

The maximum computation time of the GORPF was the longest, especially when the ESS_k value was lower than the preset threshold where parent selection, roughening, crossover and mutation were sequentially employed. The variance of the computation time spent by the GORPF was the largest because crossover and mutation were employed only at some time steps. The IPF and the proposed PF employed only crossover or mutation to find an offspring from each of the N_{kL} pairs of parents where $N_{kL} < N$ to save computation time. However, the GORPF and IPF sorted weights to find the weight threshold for parent classification, while the proposed method did not require weight sorting. The computation time of the new PF then was shorter than that of the AFPF, GORPF and IPF. However, the variance of computation time of the AFPF and that of all GA-based PF algorithms were higher than those of the SIR-PF and ASIR-PF in this experiment. The reason was that the number of the created offspring particles in the AFPF and that in GA-based PF algorithms could be different at each time step. Nevertheless, variance of computation time of the new PF was close to that of the ASIR-PF.

Figure 4.4 shows a comprehensive comparison of 1-D state estimation in terms of RMSEs where the variance of the Gaussian observation noise was $\sigma_{v_k}^2 = 2$ with different number of particles N. Figure 4.5 shows another comparison of 1-D state estimation performance in terms of average RMSEs with different levels of observation noise in signal-to-noise ratios (SNRs) where $N = 100$.

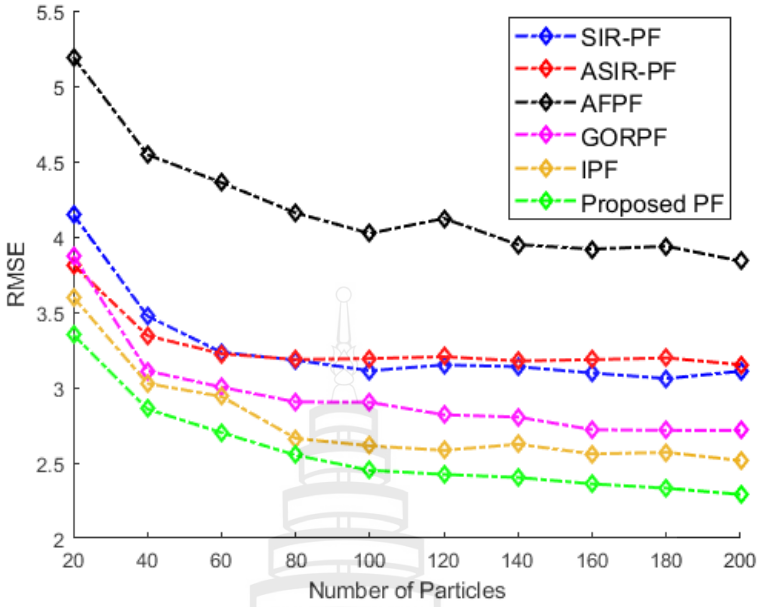


Figure 4.4 RMSEs plotted against number of particles in 1-D state estimation

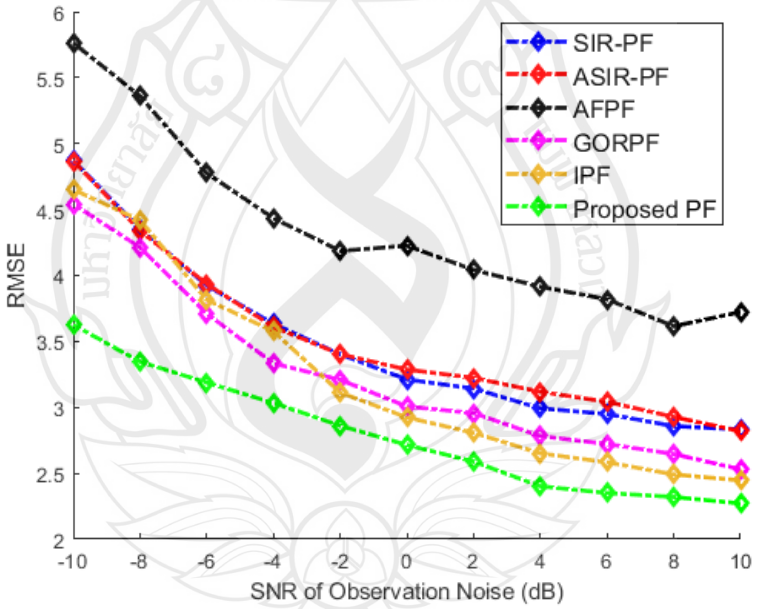


Figure 4.5 RMSEs plotted against SNRs in 1-D state estimation

The AFPF was shown to be the least effective method in this experiment as it led to the highest RMSEs for every case. The results also illustrated the importance of employing GA to find new and better particles to enhance state estimation. The trend of RMSEs of the IPF seemed more consistent than the trend of RMSEs of the GORPF when the number of particles varied as shown in Figure 4.4 and the RMSEs of the IPF

were lower than those of all non-GA-based methods (i.e., SIR-PF, ASIR-PF and AFPF). However, the trend of RMSEs against varying SNRs obtained with the GORPF seemed more consistent than the trend of RMSEs of the IPF as shown in Figure 4.5. Also, the GORPF yielded lower RMSEs compared to those of the IPF when the SNRs of the observation noise were low. Nevertheless, the new algorithm was shown to be robust for low numbers of particles (shown in Figure 4.4) and for severe observation noise (shown in Figure 4.5) by leading to the lowest RMSEs for every case.

4.2 Multidimensional State Estimation

In this section, we further test the state estimation performance on the problem of tracking the movement of a maneuvering anti-ship missile adopted from work by Zhou et al. (2019). The target state variables now become a multidimensional vector. The state evolution function and the observation function of this model are:

$$\mathbf{s}_k = \mathbf{f}_{k-1}(\mathbf{s}_{k-1}, \mathbf{u}_{k-1}) = \mathbf{\Phi}\mathbf{s}_{k-1} + \mathbf{\Gamma}\mathbf{u}_{k-1} \tag{4.7}$$

and

$$\mathbf{z}_k = \mathbf{g}_k(\mathbf{s}_k, \mathbf{v}_k) = \left[\sqrt{x_k^2 + y_k^2} \quad \arctan\left(\frac{y_k}{x_k}\right)\right]^T + \mathbf{v}_k, \tag{4.8}$$

where

$$\mathbf{s}_k = [x_k \quad y_k \quad \dot{x}_k \quad \dot{y}_k \quad \ddot{x}_k \quad \ddot{y}_k]^T \tag{4.9}$$

is the state vector of the missile that contains: the x-axis position, the y-axis position, the x-axis velocity, the y-axis velocity, the x-axis acceleration and the y-axis acceleration, respectively. Matrices $\mathbf{\Phi}$ and $\mathbf{\Gamma}$ denote the normal state evolution matrix and the state evolution noise matrix, respectively, and they are expressed as

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0 & T & 0 & \frac{T^2}{2} & 0 \\ 0 & 1 & 0 & \frac{\sin(\overline{\omega}T)}{\overline{\omega}} & 0 & \frac{1-\cos(\overline{\omega}T)}{\overline{\omega}^2} \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & \cos(\overline{\omega}T) & 0 & \frac{\sin(\overline{\omega}T)}{\overline{\omega}} \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\overline{\omega}\sin(\overline{\omega}T) & 0 & \cos(\overline{\omega}T) \end{bmatrix} \tag{4.10}$$

and

$$\mathbf{\Gamma} = \begin{bmatrix} \frac{T^3}{6} & 0 \\ 0 & \frac{\overline{\omega}T-\sin(\overline{\omega}T)}{\overline{\omega}^3} \\ \frac{T^2}{2} & 0 \\ 0 & \frac{1-\cos(\overline{\omega}T)}{\overline{\omega}^2} \\ T & 0 \\ 0 & \frac{\sin(\overline{\omega}T)}{\overline{\omega}} \end{bmatrix}, \tag{4.11}$$

where T is the sampling period (in seconds) and $\overline{\omega}$ is the maneuvering frequency (in radians). That is, the multiplication $\mathbf{\Phi s}_{k-1}$ delivers a vector of the updated x-axis position, y-axis position, x-axis velocity, y-axis velocity, x-axis acceleration and y-axis acceleration. The multiplication $\mathbf{\Gamma u}_{k-1}$ provides a vector of x-axis jolt and y-axis jolt (i.e., rate of acceleration change) that perturbs the multiplication $\mathbf{\Phi s}_{k-1}$ to compute the new state vector $\mathbf{s}_k$ as in Equation 4.7 (Zhou et al., 2019). The weight of each particle $\mathbf{s}_k^i$ in this system can be found from the likelihood function:

$$w_k^i = \exp\left(\frac{-\left(\mathbf{z}_k-\mathbf{g}_k\left(\mathbf{s}_k^i\right)\right)^T \mathbf{R}_k^{-1}\left(\mathbf{z}_k-\mathbf{g}_k\left(\mathbf{s}_k^i\right)\right)}{2}\right), \tag{4.12}$$

where $\mathbf{g}_k(\mathbf{s}_k^i)$ denotes the i-th predicted observation data value and $\mathbf{R}_k$ denotes the covariance matrix of the observation noise vector $\mathbf{v}_k$. The radar that tracks the maneuvering missile is assumed to be static and is located at the xy-position $(0, 0)$. Parameter configurations are also provided in Table 4.4.

Table 4.4 Parameters set for multidimensional state estimation experiment

Symbol	Meaning	Value
$\mathbf{s}_0$	Initial state	$[30000 \quad 3000 \quad 1450 \quad 0 \quad 0 \quad -2\pi^2]^T$
$p(\mathbf{s}_0)$	Initial prior PDF	$N(\mathbf{s}_0, 0.1\mathbf{I}_6)$
$\mathbf{u}_{k-1}$	State evolution noise	Draw from $N(\mathbf{0}, 0.1\mathbf{I}_2)$
$\mathbf{v}_k$	Observation noise	Draw from $N(\mathbf{0}, 0.01\mathbf{I}_2)$
T	Sampling period	0.1 seconds
$\bar{\omega}$	Maneuvering frequency	0.2π radians
N	Number of particles	300
K	Time steps	400
R	Simulation runs	50

In this experiment, the selection of the state inference method was also WM. There were only a few additional parameter configurations for this experiment as follows. The AFPF employs the covariance matrix $\mathbf{\Sigma} = 0.1\mathbf{I}_6$ for perturbing the state values of each created replica. The GORPF, IPF and the proposed PF also employ the covariance matrix $\mathbf{\Sigma} = 0.1\mathbf{I}_6$ in their respective mutation schemes.

The missile tracking simulation results obtained from non-GA-based PF algorithms (i.e., SIR-PF, ASIR-PF and AFPF) are shown in Figures 4.6-4.8. The missile tracking simulation results computed with the GA-based PF algorithms (i.e., GORPF, IPF and the new method) are shown in Figures 4.9-4.11. Fifty magenta curves denote curves of the estimated state and are plotted against the true state (shown as a black curve).

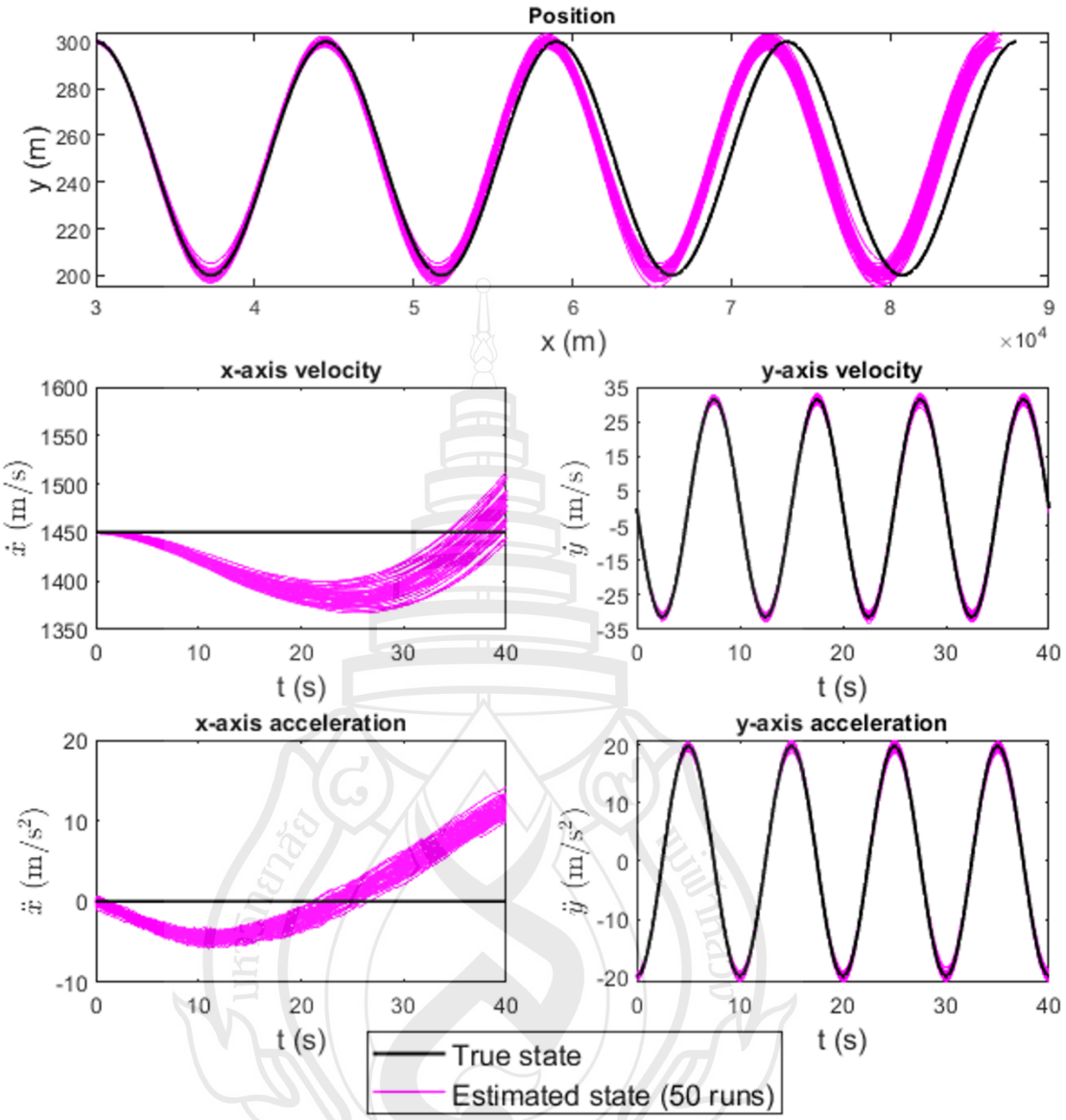


Figure 4.6 The state of the maneuvering missile tracked by the SIR-PF

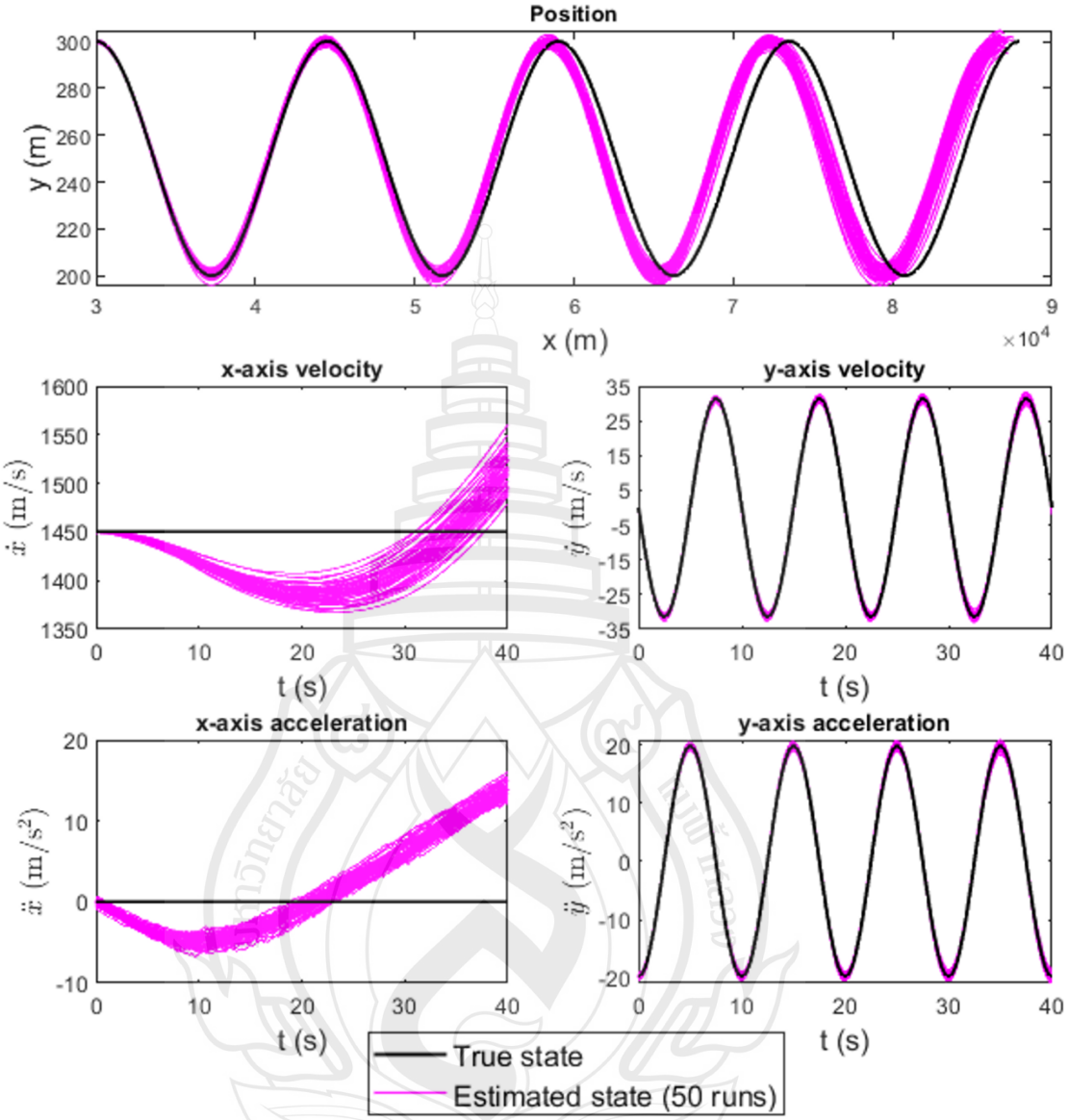


Figure 4.7 The state of the maneuvering missile tracked by the ASIR-PF

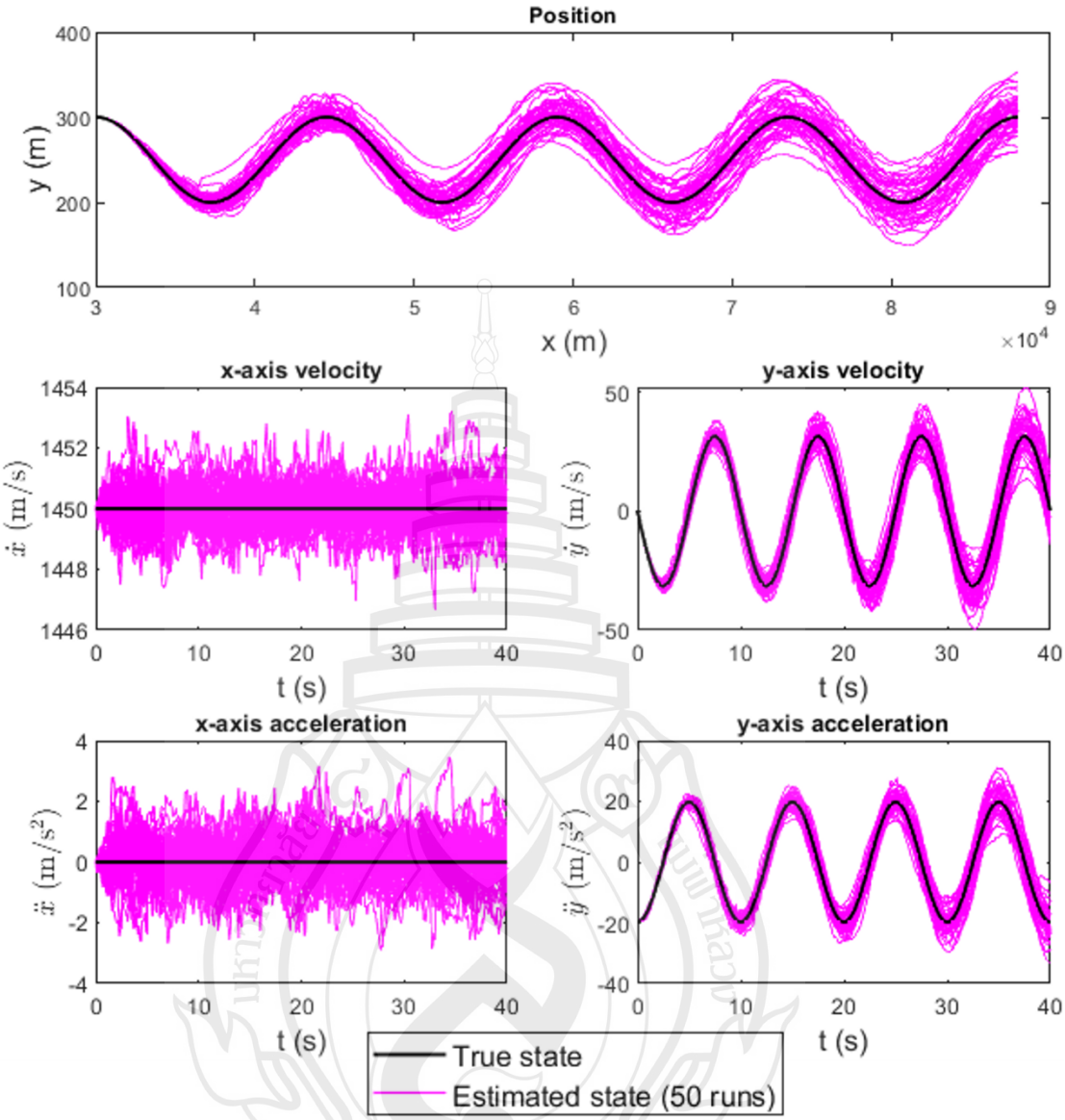


Figure 4.8 The state of the maneuvering missile tracked by the AFPF

In the SIR-PF, the x-axis acceleration was poorly estimated, and it negatively affected the x-axis velocity and position estimation as shown in Figure 4.6. The curves of the estimated x-axis acceleration kept decreasing since the beginning until the time around 10 s. Then, the estimated x-axis acceleration kept increasing and reached 0 m/s^2 at around 25 s. This caused the estimated x-axis position (or the x-axis maneuvered distance) to become shorter than the true one starting at the x-axis position of 60000 m.

The ASIR-PF mitigated the x-axis positional errors of the SIR-PF as shown in Figure 4.7. At position 80000 m on the x-axis, the curves of the estimated position

became closer to the true one. However, this was due to the overestimated x-axis accelerations that are higher than those of the SIR-PF. Thus, this position estimation cannot be considered effective. The problem was due to particle impoverishment where particles got trapped in local maxima. The AFPF led to curves of the estimated x-axis velocity and acceleration which were significantly more accurate than those of the SIR-PF and ASIR-PF, as shown in Figure 4.8. However, the AFPF resulted in the most severe errors in the y-axis state estimation. That is, the x-axis position was estimated well but most of the curves of the estimated y-axis positions clearly deviate from the true one.

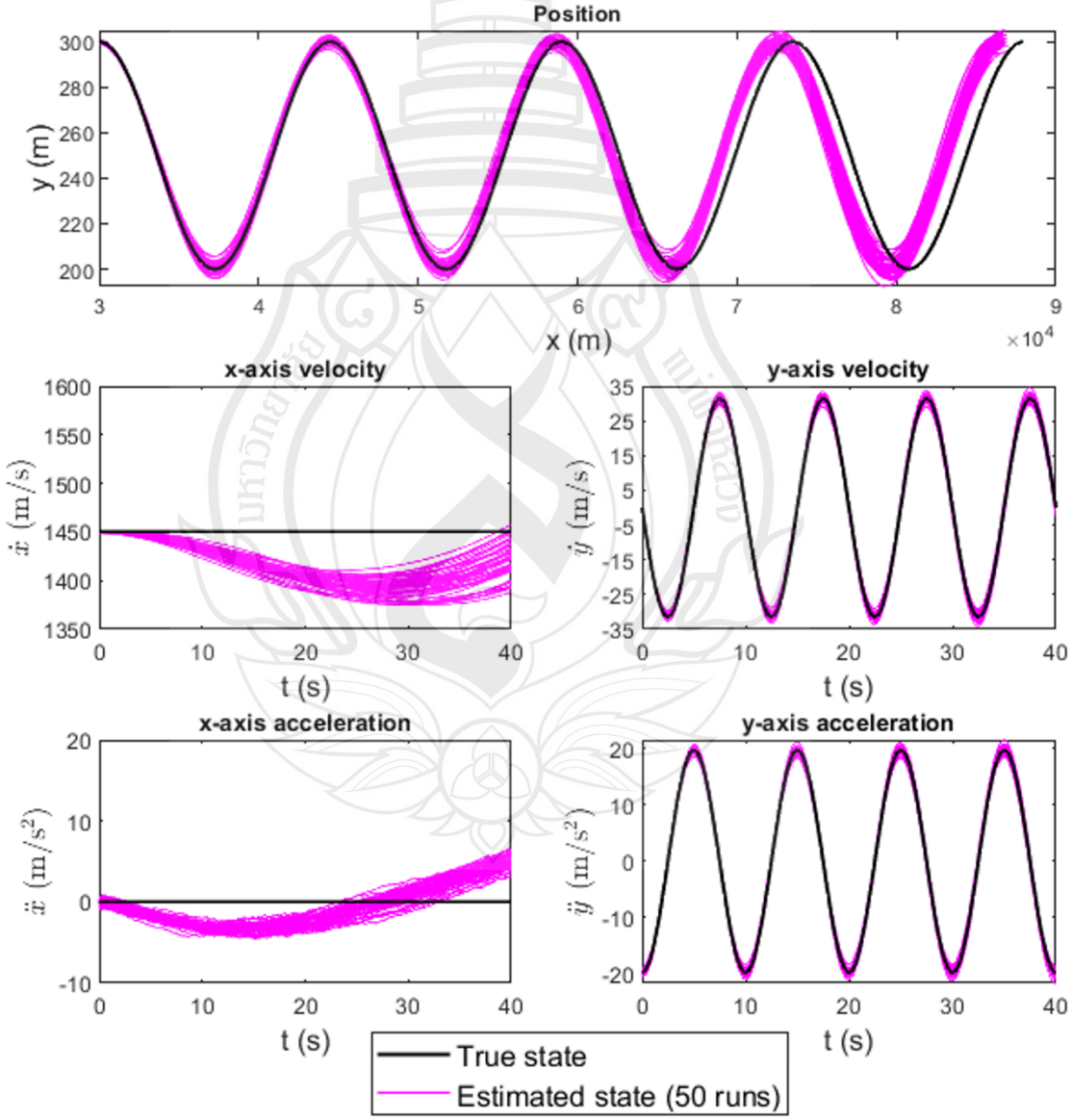


Figure 4.9 The State of the maneuvering missile tracked by the GORPF

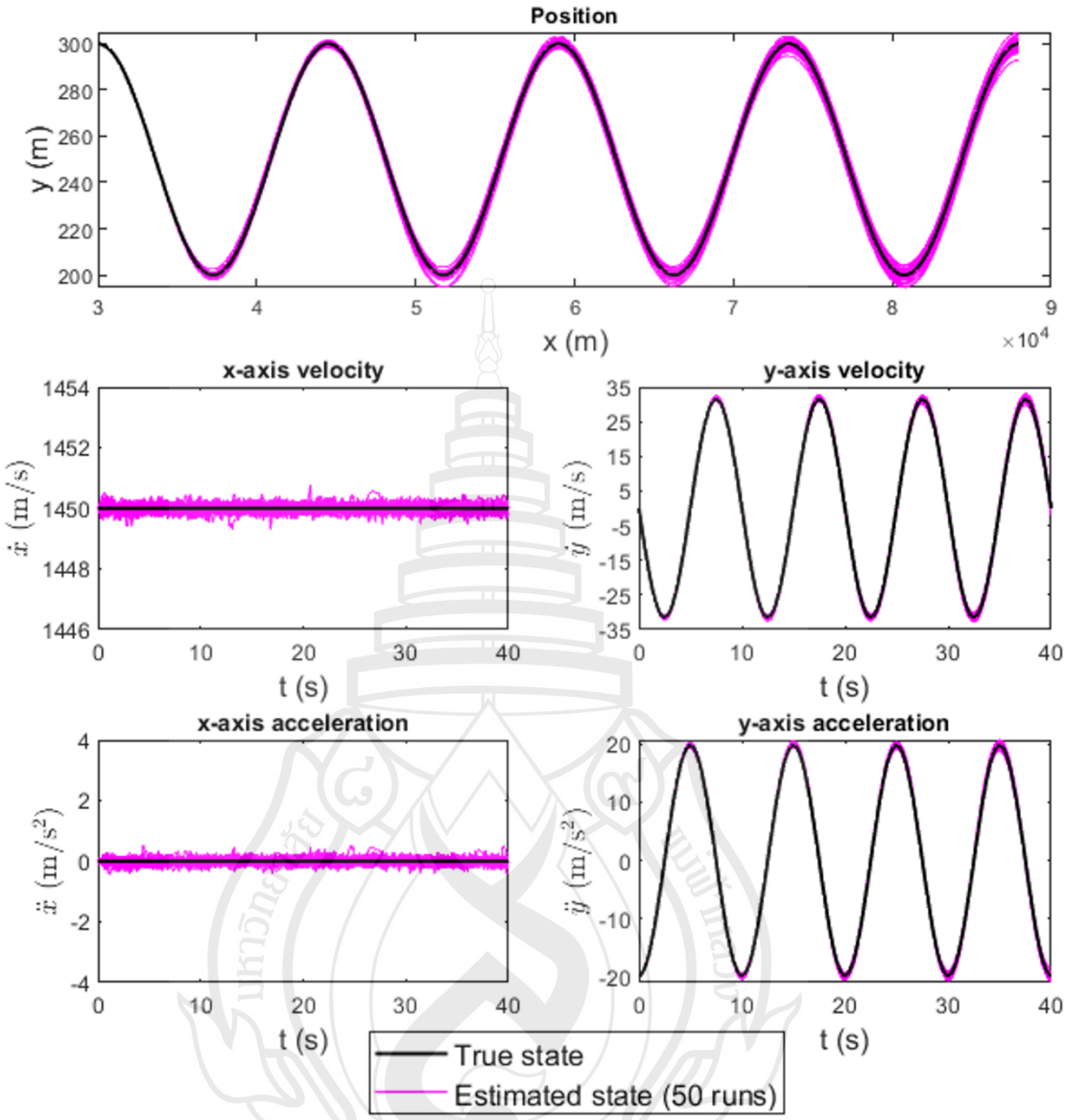


Figure 4.10 The state of the maneuvering missile tracked by the IPF

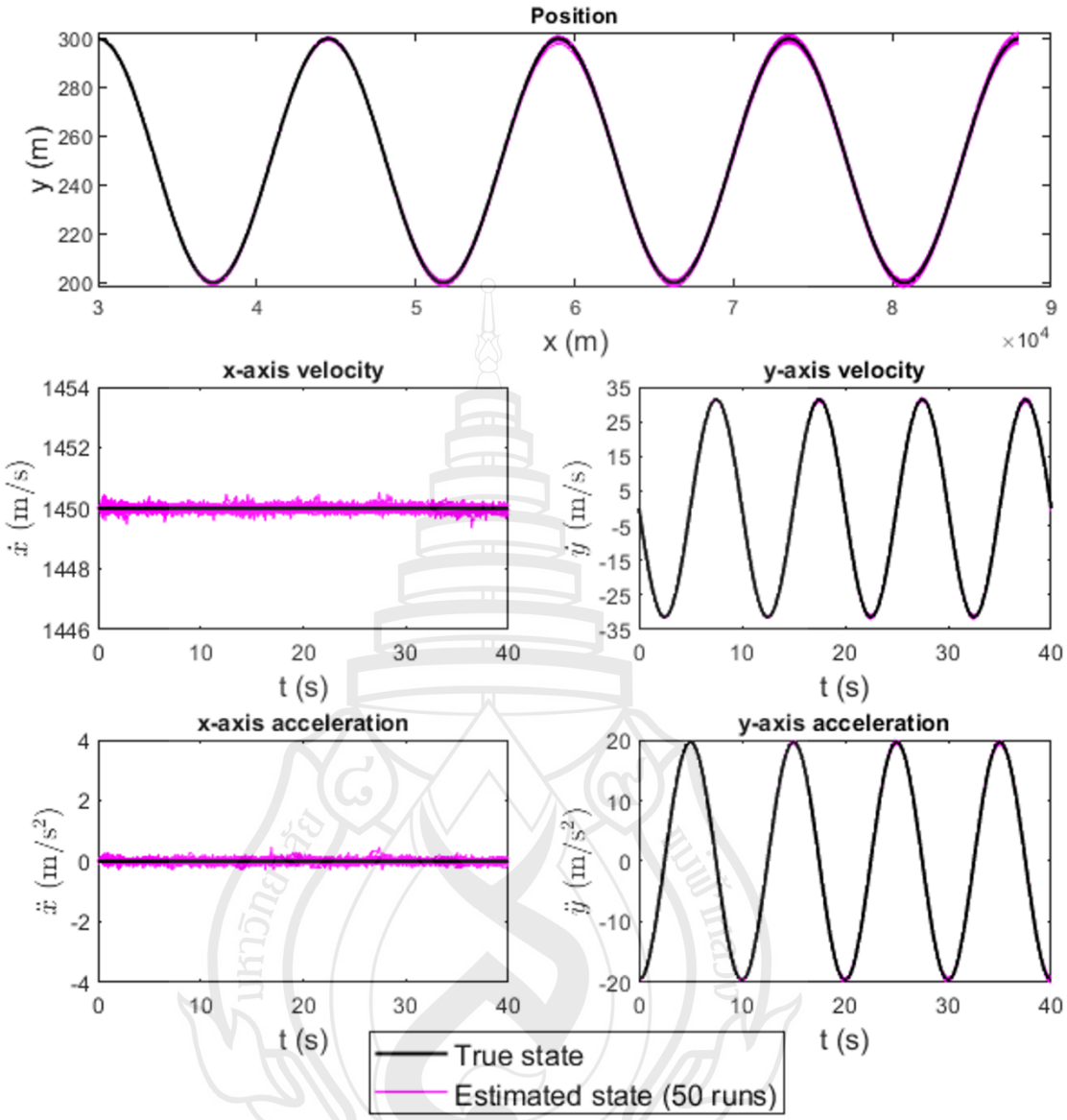


Figure 4.11 The state of the maneuvering missile tracked by the proposed PF

The GORPF faced a similar problem as the SIR-PF and ASIR-PF, providing results with erroneous estimates of x-axis positions, especially at the end of the track, as shown in Figure 4.9. Although the estimated tracks of the x-axis acceleration deviated less from the true track, the estimated x-axis accelerations were negative longer than those of the SIR-PF and ASIR-PF. Consequently, the estimated x-axis velocities were lower than the true velocity value at all times for almost every run. The estimated traveled x-axis distances of the missile were lower than the true one at the final time step (88000 m) for every run.

The IPF and the proposed method were superior to all other algorithms. While the estimated x-axis velocities and x-axis accelerations obtained from the IPF (shown in Figure 4.10) seem to not be significantly different from those computed with the new proposed method (shown in Figure 4.11), the superior accuracy of the proposed method is illustrated via the estimated position tracks. The estimated y-axis positions are clearly more accurate than those from the IPF because of more accurately estimated y-axis velocities and y-axis accelerations. Thus, our proposed method is proved more reliable than all of the other state-of-the-art algorithms.

Numerical errors were also calculated for performance evaluation of each PF algorithm in multidimensional state estimation. The average and variances of RMSEs are shown in Tables 4.5 and 4.6, respectively. The average and variances of MAEs are also shown in Tables 4.7 and 4.8 respectively.

Table 4.5 Average RMSEs in multidimensional state estimation

PF	x_k	y_k	$\dot{x}_k$	$\dot{y}_k$	$\ddot{x}_k$	$\ddot{y}_k$
SIR-PF	830.8774	1.6024	43.4959	0.5992	4.9619	0.3808
ASIR-PF	797.6756	1.5498	43.0944	0.5812	6.0826	0.3674
AFPF	0.1085	14.6709	0.6180	4.8250	0.6943	3.0276
GORPF	682.9181	2.0176	39.5444	0.7311	2.5575	0.4575
IPF	0.0655	1.4557	0.1178	0.5225	0.0941	0.3265
Proposed	**0.0625**	**0.5396**	**0.0838**	**0.2137**	**0.0621**	**0.1349**

Table 4.6 Variances of RMSEs in multidimensional state estimation

PF	x_k	y_k	$\dot{x}_k$	$\dot{y}_k$	$\ddot{x}_k$	$\ddot{y}_k$
SIR-PF	9.98e+03	0.3838	28.5326	0.0607	0.2326	0.0228
ASIR-PF	1.48e+04	0.3412	18.9508	0.0534	0.2001	0.0218
AFPF	2.13e-05	31.2837	0.0088	2.8147	0.0133	1.0635
GORPF	8.41e+03	1.0278	44.9217	0.1130	0.0598	0.0421
IPF	**1.40e-05**	0.4356	**1.00e-04**	0.0402	1.17e-04	0.0156
Proposed	3.29e-05	**0.0504**	1.08e-04	**0.0099**	**6.55e-05**	**0.0040**

Table 4.7 Average MAEs in multidimensional state estimation

PF	x_k	y_k	$\dot{x}_k$	$\dot{y}_k$	$\ddot{x}_k$	$\ddot{y}_k$
SIR-PF	638.8815	1.3520	36.8528	0.5081	4.0521	0.3221
ASIR-PF	631.8755	1.2914	36.7499	0.4965	4.8917	0.3135
AFPF	0.0869	11.6129	0.4703	3.8337	0.5396	2.3859
GORPF	504.1201	1.6742	34.0662	0.6211	2.2201	0.3880
IPF	0.0519	1.1483	0.0924	0.4166	0.0733	0.2584
Proposed	**0.0470**	**0.4319**	**0.0652**	**0.1760**	**0.0483**	**0.1107**

Table 4.8 Variances of MAEs in multidimensional state estimation

PF	x_k	y_k	$\dot{x}_k$	$\dot{y}_k$	$\ddot{x}_k$	$\ddot{y}_k$
SIR-PF	5.41e+03	0.3515	20.3666	0.0480	0.1516	0.0179
ASIR-PF	8.24e+03	0.2637	14.0046	0.0419	0.1413	0.0171
AFPF	1.72e-04	24.5280	0.0045	1.6823	0.0079	0.6363
GORPF	4.43e+03	0.8193	31.6699	0.0874	0.0479	0.0328
IPF	**1.01e-05**	0.3087	**5.08e-05**	0.0242	6.53e-05	0.0092
Proposed	2.38e-05	**0.0353**	6.65e-05	**0.0070**	**3.30e-05**	**0.0028**

The numerical results in Tables 4.5-4.8 show that the SIR-PF, the ASIR-PF and the GORPF faced severe problems in estimating the x-axis state values by having significantly high average errors and high error variances. The GORPF provided the less erroneous x-axis state estimates compared to the SIR-PF and the ASIR-PF. However, the y-axis state estimation performance of the GORPF was inferior to that of the SIR-PF and the ASIR-PF. The AFPF provided very good x-axis state estimates but the performance in estimating the y-axis state values was inferior to all other algorithms. The IPF was superior to the other state-of-the-art algorithms in terms of average errors with the exception of the proposed PF which provided the lowest average errors for every state variable. Although the variances of the errors from the proposed PF are not better (or lower) than those of the IPF for every state variable, they were quite low and were considered acceptable.

Table 4.9 provides the computation time (measured according to Equation 4.6) for each PF algorithm in multidimensional state estimation.

Table 4.9 Computation time in multidimensional state estimation

PF	Min(Time)	Avg(Time)	Max(Time)	Var(Time) ($\times 10^{-5}$)
SIR-PF	**0.0209**	**0.0274**	**0.0633**	**1.3563**
ASIR-PF	0.0313	0.0413	0.0826	3.4192
AFPF	0.0667	0.0851	0.1913	9.7741
GORPF	0.0265	0.0544	0.2191	55.9023
IPF	0.0248	0.0386	0.1661	15.9022
Proposed	0.0218	0.0319	0.0815	3.1469

In multidimensional state estimation, every PF algorithm except the SIR-PF suffered from the larger size of state vectors because each vector component (i.e., state value) of each original particle must be perturbed to find the new particles. The size of state vectors affected the SIR-PF only at the state evolution of the N particles. The SIR-PF, thus, required the shortest computation time. Although the minimum computation time of the ASIR-PF was not lower than that of the GA-based PF algorithms in this experiment, the variance of computation time of the ASIR-PF was still low. This could be due to the increased number of particles N employed in this experiment and the increased size of state vectors d_x. Recall that the ASIR-PF perturbed all of $N \times d_x$ state values of N resampled auxiliary particles, while GA-based PF algorithms needed to find only N_{KL} new offspring particles, where $N_{KL} < N$. Besides larger state vectors, the AFPF also suffered from the total number of new particles (i.e., replicas with perturbed state values) that exceeded N. Thus, the minimum and average of the AFPF computation time spent was the longest, while the maximum computation time of AFPF was slightly lower than only that of the GORPF.

Recall that the GORPF creates offspring only when the ESS_k value is lower than the preset threshold with every GA operator implemented sequentially. The maximum and variance of the computation time of the GORPF were then larger than those of the other algorithms. The minimum and average of the computation time of

the GORPF were also larger than those of the IPF and the new PF. Among non-SIR PF algorithms, the minimum and average of the computation time of the IPF were larger than only those of our method. However, the maximum computation time of IPF was larger than not only that of the new method but also of that of the ASIR-PF. Weight sorting could be a factor that caused a high maximum computation time in the AFPF, GORPF and IPF, especially when the number of particles N increased. The difference between the maximum computation time of the new method and that of the IPF then became larger. The new method was also shown in this experiment to work faster than other state-of-the art PF algorithms except for the SIR-PF, while the variance of the computation time was also significantly low.

While employing GAs increased computation time and its variance as seen in Tables 4.3 and 4.9, the numerical errors shown in Tables 4.2 and 4.5-4.8 demonstrate the importance of efforts in finding new high-weight state values that addresses the problem of a particle swarm getting trapped at a local maximum. Our method not only prevented low-quality offspring particles but also employed high-quality offspring particles as new high-weight parents. Our method was shown to be efficient and provided results that are superior to those of state-of-the-art GA-based PF algorithms, GORPF and IPF.

CHAPTER 5

APPLICATION

This chapter provides details of employing the new approach to estimate the state in a real-world application. We select the application of estimating spectra of time-varying signals in non-linear systems studied by Aunsri and Chamnongthai (2021). A broadband signal is emitted from the source located underwater (i.e., the ocean). Then, the signal propagates through the medium and is recorded by a hydrophone. We perform the signal analyses in the time-frequency domain to investigate the changes of frequencies with time in terms of the number of the modal waves (i.e., modes) that arrives at the different time and their peak frequency values (i.e., instantaneous modal frequency). Tracks of such changes contain useful information for the analysis of the dispersion characteristics of the propagation medium.

In practice, the signal received at the hydrophone can be corrupted by noise. Although the time-domain noise can be assumed to be additive white Gaussian, property of the noise that corrupts the time-frequency representation (TFR) is non-Gaussian, as to be discussed later. This necessitates the implementation of PFs for the modal frequency estimation. The accuracy of the modal frequency estimation affects the validity of further research work related to the environmental studies.

5.1 Time-frequency Analysis of Underwater Broadband Signals

In theory, we calculate the sound pressure against time of a broadband time-series that propagates in the ocean as:

$$p(r, d_s, d_r, t) = \frac{1}{2\pi}\sum_m \int \mu(\omega')G_m(r, d_s, d_r, \omega')\exp\left\{j\left(\omega' t - k_m r - \frac{\pi}{4}\right)\right\} d\omega', \tag{5.1}$$

where quantity r denotes the distance between the source and the receiver (i.e., hydrophone), quantities d_s and d_r denote the source and receiver depths, respectively,

quantity k_m denotes the modal wave number, $\mu(\omega')$ denotes the source spectrum, $\omega = 2\pi f$ denotes the angular frequency in radians per second (rad/s), and f denotes the frequency in Hertz (Hz) (Aunsri & Chamnongthai, 2021; Yang, 1984). Function $G_n(r, d_s, d_r, \omega')$ is the mode transfer function that can be expressed as:

$$G_m(r, d_s, d_r, \omega) = \frac{j\sqrt{\pi}}{\rho(d_r)\sqrt{2k_m r}} \Psi_m(d_s)\Psi_m(d_r), \tag{5.2}$$

where $\Psi_m(\cdot)$ denotes the orthogonal and normalized depth-dependent functions, and $\rho(d_r)$ denotes the medium density. When the signal has only one mode (i.e., the m-th mode), the spectrum of a finite segment of such a signal can be computed as:

$$P_m(\omega, t) = \int_{t-\Delta t}^{t+\Delta t} p_m(r, d_s, d_r, \tau)\exp(-j\omega\tau)d\tau, \tag{5.3}$$

which starts and ends at time $t - \Delta t$ and $t + \Delta t$, respectively. Consequently, we obtain

$$P_m(\omega, t) = \frac{\exp(-j\omega t)}{2\pi} \int \mu(\omega')G_m(r, d_s, d_r, \omega') \frac{\sin(\omega' - \omega)\,\Delta t}{(\omega' - \omega)} \times \exp\left\{j\left(\omega' t - k_m r - \frac{\pi}{4}\right)\right\} d\omega', \tag{5.4}$$

whose instantaneous power spectrum can be obtained by squaring the spectrum approximated via stationary phase approximation as:

$$|P_m(\omega, t)|^2 = \frac{\pi}{\left|k_m''\right|^2} |\mu(\omega_m)G_m(r, d_s, d_r, \omega_m)|^2 \left|\frac{\sin(\omega-\omega_m)\Delta t}{(\omega-\omega_m)}\right|^2$$

$$|P_m(\omega, t)|^2 = \frac{\pi(\Delta t)^2}{\left|k_m''\right|^2} |\mu(\omega_m)G_m(r, d_s, d_r, \omega_m)|^2 \left|\frac{\sin(\omega-\omega_m)\Delta t}{(\omega-\omega_m)\Delta t}\right|^2, \tag{5.5}$$

for $|\omega - \omega_m| < \pi/\Delta t$. That is, the power spectrum of the m-th single-mode signal in Equation 5.5 has a peak at the angular frequency ω_m which is regarded as the instantaneous modal frequency. Consequently, the power spectrum can be approximated as a summation of squared sinc pulses (Aunsri & Chamnongthai, 2021; Yang, 1984).

In this experiment, we employ a short-time Fourier transform (STFT) to compute the TFR of a time-varying input signal. The reason is that the STFT does not introduce false frequency modes called “cross-terms” which negatively affect the readability of the TFR (Boashash, 2016). Suppose that we have an input signal:

$$x(\tau) = s(\tau) + n(\tau), \tag{5.6}$$

where $s(\tau)$ and $n(\tau)$ represent a noise-free signal and a series of additive white Gaussian noise values. We compute the STFT of the signal $x(\tau)$ and time t and frequency f as:

$$\begin{aligned} STFT_x(t,f) &= \int x(\tau)w(t-\tau)\exp(-j2\pi f\tau)\,d\tau \\ STFT_x(t,f) &= \int [s(\tau)+n(\tau)]w(t-\tau)\exp(-j2\pi f\tau)\,d\tau \\ STFT_x(t,f) &= STFT_s(t,f) + STFT_n(t,f), \end{aligned} \tag{5.7}$$

where $w(\tau)$ denotes a window function employed in the STFT calculation. Terms $STFT_s(t,f)$ and $STFT_n(t,f)$ represent the STFT of the noise-free signal $s(\tau)$ and the STFT of the noise $n(\tau)$, respectively. Sometimes we can find the term $w(\tau - t)$ written as $w(t-\tau)$ in Equation 5.7 because the employed window function $w(\tau)$ is normally an even function where $w(\tau) = w(-\tau)$ (Boashash, 2016).

For simplicity, we assume that the term $STFT_x(t,f)$ is a Gaussian random variable:

$$STFT_x(t,f) \sim N(STFT_s(t,f), \sigma^2), \tag{5.8}$$

where σ^2 is an unknown variance. Because $STFT_x(t,f)$ is a complex number, we can then assume that the real part:

$$\mathrm{Re}(STFT_x(t,f)) = \int x(\tau)w(t-\tau)\cos(-j2\pi f\tau)\,d\tau \tag{5.9}$$

and the imaginary part:

$$\mathrm{Im}(STFT_x(t,f)) = \int x(\tau)w(t-\tau)\sin(-j2\pi f\tau)\,d\tau, \tag{5.10}$$

are corrupted by additive white Gaussian noise with the same variance at $\sigma^2/2$. That is,

$$\mathrm{Re}(STFT_x(t,f)) \sim N(\mathrm{Re}(STFT_s(t,f)), \sigma^2/2), \tag{5.11}$$

and

$$\mathrm{Im}(STFT_x(t,f)) \sim N(\mathrm{Im}(STFT_s(t,f)), \sigma^2/2). \tag{5.12}$$

The spectrogram at time t and frequency f of the signal $x(\tau)$ can be found as the squared magnitude of its STFT:

$$SP_x(t,f) = \left| \int x(\tau) w(t-\tau) \exp(-j2\pi f\tau)\, d\tau \right|^2$$

$$SP_x(t,f) = \left[\mathrm{Re}\left(STFT_x(t,f)\right)\right]^2 + \left[\mathrm{Im}\left(STFT_x(t,f)\right)\right]^2, \tag{5.13}$$

where the PDF of the squared STFT, $SP_s(t,f)$, becomes a noncentral chi-squared PDF with two degrees of freedom; the parts of the noise-free STFT (i.e., $\mathrm{Re}(STFT_s(t,f))$ and $\mathrm{Im}(STFT_s(t,f))$ in Equations 5.11-5.12) are not necessarily zero for each time t and frequency f (Aunsri & Chamnongthai, 2021; Boashash, 2016).

In practice, any input signal is discrete because it is recorded with a preset sampling rate. The spectrogram of such a signal at time step k and frequency f can be found as:

$$SP_x(k,f) = \left[\mathrm{Re}\left(STFT_x(k,f)\right)\right]^2 + \left[\mathrm{Im}\left(STFT_x(k,f)\right)\right]^2, \tag{5.14}$$

where

$$\mathrm{Re}\left(STFT_x(k,f)\right) = \sum_{l=0}^{L_w-1} x[l] w[k-l] \cos\left(-2\pi f \frac{l}{L_{DFT(x)}}\right) \tag{5.15}$$

and

$$\mathrm{Im}\left(STFT_x(k,f)\right) = \sum_{l=0}^{L_w-1} x[l] w[k-l] \sin\left(-2\pi f \frac{l}{L_{DFT(x)}}\right), \tag{5.16}$$

are the real part and imaginary parts of the STFT at time step k and frequency f, respectively (Aunsri, 2019; Huillery et al., 2008; Kuptametee & Aunsri, 2022c; Tan & Jiang, 2019). Quantity l denotes time, while $L_{DFT(x)}$ and L_w represent the length of the

discrete Fourier transform (DFT) of the input signal $x[l]$ and the length of the employed window function $w[l]$, respectively.

5.2 Particle Filtering Formulation for Spectra Estimation

5.2.1 PF Initialization

Because we do not know the true states at the initialization, we can draw particles to predict the state vector $\mathbf{x}_1$ and find their weights according to the first observation $\mathbf{y}_1$. Note that this application differs from the experiments presented in Chapter 4 where the initial state vector $\mathbf{x}_0$ was known and employed to initialize the state evolution function.

In case of the first time step (i.e., $k = 1$), we need to draw the number of modes r_1^i for each i-th initial particle $\mathbf{x}_1^i$ as:

$$r_1^i \sim U\{r_{min}, r_{max}\}, \tag{5.17}$$

where $U\{r_{min}, r_{max}\}$ denotes a discrete uniform distribution. Quantities r_{min} and r_{max} are the minimum and maximum of the number of modes, respectively.

Next, we draw r_1^i modal frequencies for each initial particle $\mathbf{x}_1^i$ and store them as a state vector $\mathbf{f}_1^i$. Each m-th modal frequency where $m \in \{1, \dots, r_1^i\}$ can be drawn as:

$$f_{1,m}^i \sim U\{f_{min}, f_{max}\}, \tag{5.18}$$

where $f_{1,m}^i$ denotes the modal frequency of the m-th mode of the i-th initial particle $\mathbf{x}_1^i$. Quantities f_{min} and f_{max} are a preset minimum modal frequency and a preset maximum modal frequency, respectively.

Next, we set the initial prior PDF of the modal amplitude (i.e., peak amplitude) for each initial particle $\mathbf{x}_1^i$ and store them as a state vector $\mathbf{a}_1^i$. Each m-th modal amplitude where $m \in \{1, \dots, r_1^i\}$ can be drawn as a positive real-numbered value:

$$a_{1,m}^i \sim U(0, \max(\mathbf{y}_1)], \tag{5.19}$$

where $\max(\mathbf{y}_1)$ denotes the maximum amplitude value searched through the observation $\mathbf{y}_1$.

Also, we draw the initial STFT noise variance value for each particle initial particle $\mathbf{x}_1^i$ as:

$$\sigma_1^{2,i} \sim U(0, \sigma_{max}^2], \tag{5.20}$$

where σ_{max}^2 denotes the preset upper bound of the noise variance values; the noise variance must be greater than zero.

5.2.2 State Vector Evolution

At time step $k \geq 2$, we update the modal frequencies of each particle as:

$$\mathbf{f}_k^i \sim N(\mathbf{f}_{k-1}^i, \mathbf{\Sigma}_{f,k-1}), \tag{5.21}$$

where $\mathbf{\Sigma}_{f,k-1}$ denotes a covariance matrix with the dimension $r_{k-1}^i \times r_{k-1}^i$ employed to update the frequency values.

Next, we update modal amplitudes for each particle $\mathbf{x}_k^i$ as:

$$\mathbf{a}_k^i \sim N(\mathbf{a}_{k-1}^i, \mathbf{\Sigma}_{a,k-1}), \tag{5.22}$$

where $\mathbf{\Sigma}_{a,k-1}$ denotes a covariance matrix with the dimension $r_{k-1}^i \times r_{k-1}^i$ employed to update the amplitude values.

Next, we update the noise variance value of each particle $\mathbf{x}_k^i$ as:

$$\sigma_k^{2,i} \sim N(\sigma_{k-1}^{2,i}, \zeta^2), \tag{5.23}$$

where ζ^2 denotes the preset variance of the noise variance updating function.

Recall that the number of modes can vary with time because each mode arrives the receiver at the different time. The size of each vector $\mathbf{f}_k^i$ and the size of each vector $\mathbf{a}_k^i$ (i.e., number of modes r_k^i) must then be further updated. The state evolution of modal frequencies is then encompassed within the multiple-model particle filter (MMPF) framework (Aunsri & Michalopoulou, 2014; Aunsri, 2018b). In this work, we assume that the number of modes can stay the same, can decrease by one, or can increase by

one. The transition matrix of the probabilities of change of the number of modes can then be employed and expressed as:

$$P = \begin{bmatrix} p & 1-p & 0 \\ (1-p)/2 & p & (1-p)/2 \\ 0 & 1-p & p \end{bmatrix}, \tag{5.24}$$

where p denotes the probability that the number of modes will remain the same (i.e., $r_k^i = r_{k-1}^i$); $0 \leq p \leq 1$. The probabilities in the first row of the matrix P are employed when the number of modes of a particle is at the minimum; the number of modes will increase by one with the probability $1-p$. The probabilities in the second row of the matrix P are employed when the number of modes of a particle is at neither the minimum nor the maximum; the probability that the number of modes will decrease by one and the probability that the number of modes will increase by one are the same at $(1-p)/2$. The probabilities in the third row of the matrix P are employed when the number of modes of a particle is at the maximum; the number of modes will decrease by one with the probability $1-p$ (Aunsri & Michalopoulou, 2014).

When $r_k^i = r_{k-1}^i + 1$, we can simply draw the $(r_{k-1}^i + 1)$-th new modal frequency as:

$$f_{k,r_{k-1}^i+1}^i \sim U\{f_{min}, f_{max}\}, \tag{5.25}$$

which can be appended to the vector $\mathbf{f}_k^i$ previously found via Equation 5.21. The modal amplitude of the newly added mode can also be drawn as a positive real-numbered value:

$$a_{k,r_{k-1}^i+1}^i \sim U[\min(\mathbf{a}_{k-1}^i), \max(\mathbf{a}_{k-1}^i)], \tag{5.26}$$

which can be appended to the vector $\mathbf{a}_k^i$ previously found via Equation 5.22. Quantities $\min(\mathbf{a}_{k-1}^i)$ and $\max(\mathbf{a}_{k-1}^i)$ denote the minimum and the maximum amplitude value of the particle $\mathbf{x}_{k-1}^i$.

When $r_k^i = r_{k-1}^i - 1$, we can simply remove the r_{k-1}^i-th modal frequency and the r_{k-1}^i-th modal amplitude from the vectors $\mathbf{f}_k^i$ and $\mathbf{a}_k^i$ previously found via Equations 5.21 and 5.22, respectively.

5.2.3 Particle Weights Calculation

Let we first consider a noncentral chi-squared PDF of a 1-D random variable $z \sim \sum_{i=1}^{df} x_i^2$; quantity df denotes the degree of freedom and $x_i \sim N(\mu_i, \sigma^2)$. Such a PDF can be expressed as[16,62]:

$$f(z; df, \lambda) = \frac{1}{2\sigma^2}\left(\frac{z}{\lambda}\right)^{(df-2)/4} \exp\left(-\frac{(z+\lambda)}{2\sigma^2}\right) I_{(df-2)/2}\left(\frac{\sqrt{z\lambda}}{\sigma^2}\right), \quad (5.27)$$

where

$$\lambda = \sum_{i=1}^{df} \mu_i^2, \quad (5.28)$$

denotes the noncentrality parameter. $I_n(\cdot)$ denotes the n-th order modified Bessel function of the first kind.

To calculate weight of each particle at each time step k, we must first construct the spectrum replica of each particle as:

$$\boldsymbol{s}_k^i = \sum_{m=1}^{r_k^i} a_{k,m}^i \mathrm{sinc}^2(f - f_{k,m}^i), \quad (5.29)$$

where quantity $a_{k,m}^i$ denotes the modal amplitude of the m-th mode (i.e., the m-th squared sinc function) at time step k. Note that, when there is only one mode (i.e., $r_k^i = 1$), Equation 5.29 can be considered as a simplification of Equation 5.5.

The prior PDF of the number of modal frequencies is $p(r_k) = 1/(r_{max} - r_{min} + 1)$. That is, the probability is the same for every value of r_k, where $r_k \in \{r_{min}, \dots, r_{max}\}$. The prior PDF of each modal frequency is $p(f_{k,m}) = 1/L$; quantity L represents the length of the spectrum. If a particle has r_k modes, the prior PDF can be generalized as $p(f_{k,m}, r_k) = 1/L^{r_k}$. The prior PDF of each modal amplitude is non-informative because each modal amplitude can be any non-negative real number. Recall that the magnitude the spectrogram at time step k and frequency f, $SP_x(k,f)$, in Equation 5.14 can be considered as a noncentral chi-squared random variable with two degrees of freedom and variance at $\sigma^2/2$. The likelihood function can then be expressed as:

$$w_k^i = \frac{1}{L^{r_k^i}} \frac{1}{\left(\sigma_k^{2,i}\right)^L} \prod_{f=1}^{L} \exp\left(-\frac{(y_{k,f}+s_{k,f}^i)}{\sigma_k^{2,i}}\right) I_0\left(\frac{\sqrt{y_{k,f} \times s_{k,f}^i}}{\sigma_k^{2,i}/2}\right), \quad (5.30)$$

where quantities $y_{k,f}$ and $s_{k,f}^i$ represent the magnitude at time step k and frequency f of the observation and that of the spectrum replica, respectively (Aunsri & Chamnongthai, 2021).

5.2.4 Employment of Our Proposed Method

Because the number of modes can vary with time, the size of each state vector (i.e., particle) can be different. Suppose that each l-th low-weight parent $\mathbf{x}_{kL}^l$ is paired with the randomly selected h-th high-weight parent $\mathbf{x}_{kH}^h$, where $l \in \{1, \ldots, N_{kL}\}$, $h \sim U\{1, \ldots, N_{kH}\}$, $N_{kL} < N$, and $N_{kH} < N$. Quantities r_{kH}^h and r_{kL}^l denote the number of modes of the high-weight parent $\mathbf{x}_{kH}^h$ and that of the low-weight parent $\mathbf{x}_{kL}^l$, respectively. Only one of two GA operators (i.e., flat crossover proposed by Radcliffe (1990) or modified Gaussian mutation proposed by Zhang et al. (2021)) is randomly chosen according to quantity ESS_k of all N pre-classification parents (i.e., parameter γ_k calculated via Equation 3.2) and employed to calculate only new modal frequencies and new modal amplitudes for each offspring. Regardless of the choice of GA operators, each offspring must inherit the noise variance value $\sigma_k^{2,i}$ of its own low-weight parent $\mathbf{x}_{kL}^l$ without perturbing it. The reason is to obtain fair comparison of the weight of the spectrum replica generated from state values of the offspring particle and the weight of the spectrum replica generated from state values of the low-weight parent with the same noise variance value.

In flat crossover, only modal frequencies and modal amplitudes of the first through the $\min(r_{kH}^h, r_{kL}^l)$-th mode of the high-weight parent $\mathbf{x}_{kH}^h$ and those of the low-weight parent $\mathbf{x}_{kL}^l$ are employed to calculate offspring state values. In case $r_{kL}^l > r_{kH}^h$, modal frequencies and modal amplitudes of the $(r_{kH}^h + 1)$-th through the r_{kL}^l-th mode of the low-weight parent $\mathbf{x}_{kL}^l$ will be assigned to the offspring without being changed. In case $r_{kL}^l < r_{kH}^h$, r_{kL}^l modal frequencies and r_{kL}^l modal amplitudes of the new offspring can be completely different from those of its two parents. Modal frequencies and modal

amplitudes of the $(r_{kL}^{l}+1)$-th through the r_{kH}^{h}-th mode of the high-weight parent $\mathbf{x}_{kH}^{h}$ are left unpaired and unemployed in this case.

When modified Gaussian mutation is employed, the size of the offspring vector must be same as that of its high-weight parent $\mathbf{x}_{kH}^{h}$. That is, the offspring in this case can be found by: (1) creating a copy of its low-weight parent $\mathbf{x}_{kL}^{l}$, (2) replacing the old modal frequencies and modal amplitudes with those of its high-weight parent $\mathbf{x}_{kH}^{h}$, and (3) perturbing the newly assigned modal frequencies and modal amplitudes.

After new modal frequencies and new modal amplitudes are obtained for an offspring, we need to find the new spectrum replica and the new weight of that offspring (with the inherited unchanged noise variance $\sigma_k^{2,i}$) according to Equations 5.29-5.30. We accept the new offspring only if its weight is higher than that of its low-weight parent $\mathbf{x}_{kL}^{l}$. Recall that, before the first offspring is found at any time step, $N_{kL}+N_{kH}=N$. Because our method suggests that any offspring particle whose weight is not lower than the weight threshold employed to classify the N parents should be employed as a new high-weight parent, quantity N_{kH} must be increased by one every time such an offspring is found.

5.2.5 Spectra Estimation

After GA is employed, some original high-weight parents may have weights that are lower than those of their respective offspring. In this application, we also perform resampling to eliminate low-weight particles and to ensure the existence of high-weight particles.

Suppose that the resampled particle $\tilde{\mathbf{x}}_k^i$ has $\tilde{r}_k^i$ modes where $r_{min} \le \tilde{r}_k^i < r_{max}$. We select the most frequently obtained number of modes as the number of modes of the estimated spectrum to be created at time step k:

$$\hat{r}_k = \underset{i}{\mathrm{MAP}}(\tilde{r}_k^i). \tag{5.31}$$

Next, we find the most frequently obtained non-zero modal frequency of each mode at time step k:

$$\hat{f}_{k.m} = \underset{i}{\mathrm{MAP}}(\tilde{f}_{k,m}^i), \tag{5.32}$$

where $m \in \{1, \ldots, \hat{r}_k\}$.

Next, we find the modal amplitude of each estimated modal frequency $\hat{f}_{k.m}$ at time step k. In this step, we first need to find resampled particles $\tilde{\mathbf{x}}_k^i$ whose modal frequencies $\tilde{f}_{k,m}^i$ are equal to $\hat{f}_{k,m}$ (computed via Equation 5.32). Let $N_{\hat{f}_{k,m}}$ denote the number of resampled particles whose modal frequencies $\tilde{f}_{k,m}^i$ satisfy the condition $\tilde{f}_{k,m}^i = \hat{f}_{k,m}$, where $N_{\hat{f}_{k,m}} \leq N$; quantity $N_{\hat{f}_{k,m}}$ can be different for each m-th mode. The m-th modal amplitude $\tilde{a}_{k,m}^i$ of such resampled particles is kept unchanged, while those of the rest are set as zero, because modal amplitudes $\tilde{a}_{k,m}^i$ of resampled particles whose modal frequencies $\tilde{f}_{k,m}^i$ do not satisfy the condition $\tilde{f}_{k,m}^i = \hat{f}_{k,m}$ will be excluded. Then, we can find each m-th modal amplitude of the estimated spectrum at time step k as

$$\hat{a}_{k.m} = \sum_{i=1}^{N} \frac{\tilde{a}_{k,m}^i}{N_{\hat{f}_{k,m}}}, \tag{5.33}$$

which is equivalent to finding an average of the m-th modal amplitude $\tilde{a}_{k,m}^i$ that belongs to the $N_{\hat{f}_{k,m}}$ resampled particles. That is, the denominator in Equation 5.33 must be $N_{\hat{f}_{k,m}}$ instead of N, while quantity $\tilde{a}_{k,m}^i$ of each unused resampled particle has already been set to zero.

Finally, we create the estimated spectrum at time step k as

$$\hat{\mathbf{s}}_k = \sum_{m=1}^{\hat{r}_k} \hat{a}_{k,m} \mathrm{sinc}^2(f - \hat{f}_{k,m}) \tag{5.34}$$

which is a summation of $\hat{r}_k$ squared sinc functions with their respective modal frequencies $\hat{f}_{k,m}$ (found via Equation 5.32) and respective modal amplitudes $\hat{a}_{k,m}$ (found via Equation 5.33).

5.3 Experimental Results

A time-varying broadband signal emitted and from a sound source received at a hydrophone in the ocean (shown in Figure 5.1) is chosen as the input signal. The

sampling rate employed in recording the signal was 2000 Hz. A spectrogram of the signal is also shown in Figure 5.2. We selected a Hamming window as suggested by Aunsri (2018a) with a length at 180 milliseconds (ms). However, we choose to perform state estimation only between 451 to 1050 ms of the spectrogram where the dispersion curves (i.e., tracks of the modal waves) seem well-separated as shown in Figure 5.3. Figure 5.4 shows the zoomed version of such a portion of the spectrogram at frequency 200 to 600 Hz.

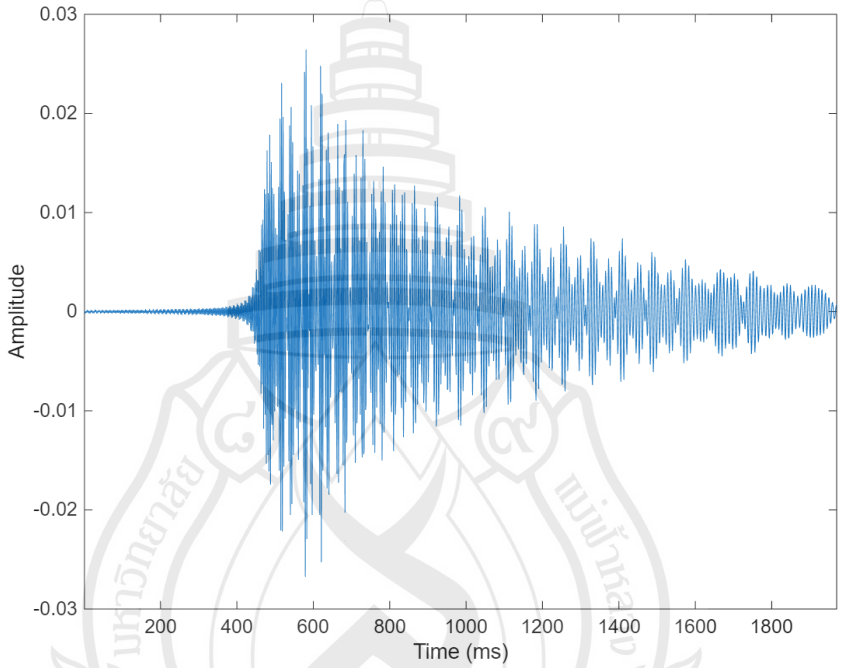


Figure 5.1 A noise-free acoustic time-series

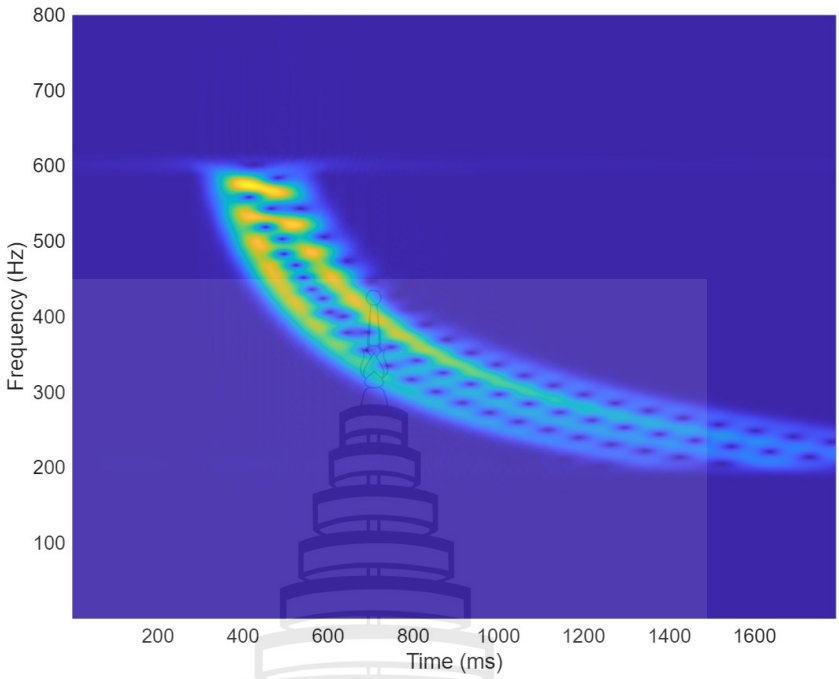


Figure 5.2 A spectrogram of the noise-free acoustic time-series

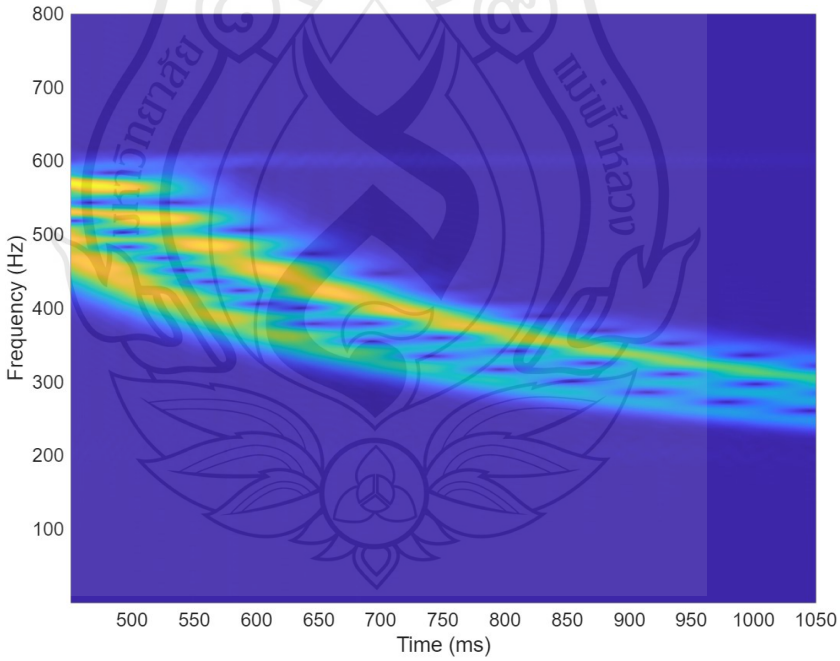


Figure 5.3 A portion of the spectrogram

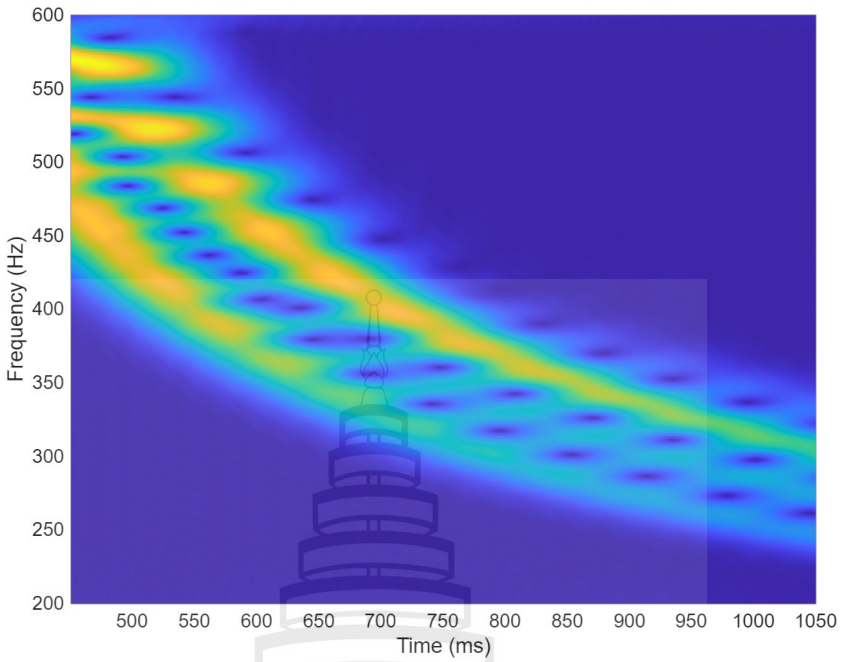


Figure 5.4 A zoomed portion of the spectrogram

The spectrum estimation performance of our method will be compared with that of the traditional sequential importance resampling particle filter (SIR-PF) and the percentile-based resampling particle filter (PBR-PF) proposed by Aunsri et al. (2021). According to the PF formulation discussed in Section 5.2, at each time step, the SIR-PF and PBR-PF perform Steps 5.2.1-5.2.3 and then skip to Step 5.2.5 because these PF algorithms do not employ GA. At the beginning of Step 5.2.5, the SIR-PF and the proposed method employ a systematic resampling scheme. The PBR-PF, however, keeps and replicates only the $N_{PBR,k}$ best particles with the summation of their weights not less 90% of the summation of all N weights at time step k, while the rest are eliminated (Aunsri et al., 2021). Recall that $N_{PBR,k} < N$ and $N_{PBR,k}$ must be the lowest integer that satisfies such a condition and preset percentage. Also, the weights of the selected and sorted $N_{PBR,k}$ particles must first be normalized according to Equation 2.34. Figure 5.5 shows a squared sinc function employed in this experiment to calculate the replica of the spectrum according to Equation 5.29.

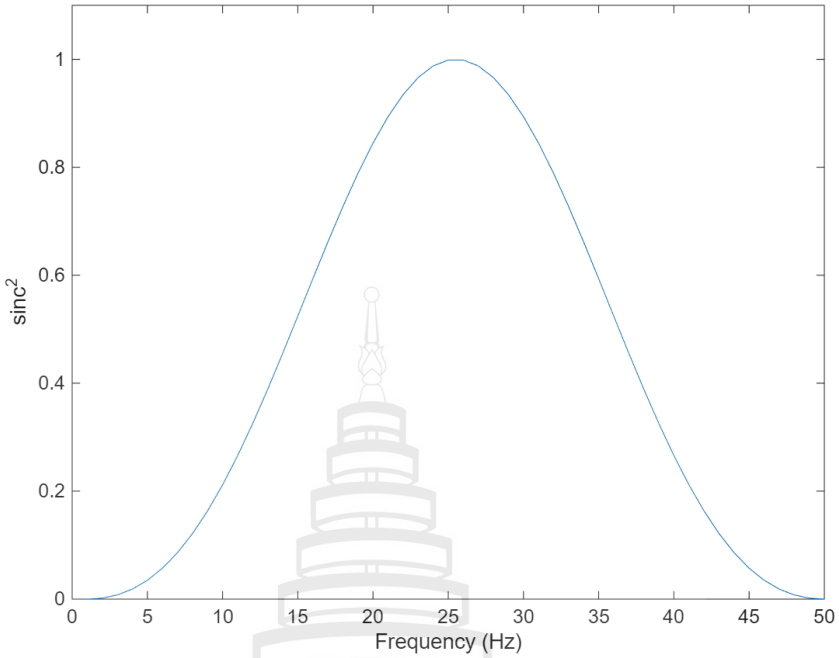


Figure 5.5 A squared sinc function used in spectrum estimation

Table 5.1 presents the related parameters preset for the PF algorithms. The proposed method employs modified Gaussian mutation with covariance matrices $\mathbf{\Sigma}_f = \mathbf{I}_{r_{kH}^h}$ and $\mathbf{\Sigma}_a = 10^{-3}\mathbf{I}_{r_{kH}^h}$ to find new modal frequencies and new modal amplitudes, respectively. Matrix $\mathbf{I}_{r_{kH}^h}$ denotes an identity matrix with dimension $r_{kH}^h \times r_{kH}^h$ where r_{kH}^h is the number of modes of the high-weight parent $\mathbf{x}_{kH}^h$.

Table 5.1 Parameters used in spectrum estimation

Symbol	Meaning	Value
L	Length of spectrum	800
r_{min}	Minimum number of modes	2
r_{max}	Maximum number of modes	6
f_{min}	Minimum modal frequency	200
f_{max}	Maximum modal frequency	600
σ_{max}^2	Maximum noise variance value	10^{-4}
t_{first}	First time step (ms)	451
t_{last}	Last time step (ms)	1050
p	Probability that variable r_{k-1} stays unchanged	0.6
$\mathbf{\Sigma}_{f,k-1}$	Covariance matrix of PDF of modal frequencies	$\mathbf{I}_{r_{k-1}}$
$\mathbf{\Sigma}_{a,k-1}$	Covariance matrix of PDF of modal amplitudes	$10^{-3}\mathbf{I}_{r_{k-1}}$
ζ^2	Variance of PDF of noise variance	10^{-6}
N	Number of particles	2000
K	Number of time steps	600
R	Number of simulation runs	50

In this experiment, the time-domain observation noise is additive white Gaussian with a fixed but unknown variance. The SNR for each case, however, cannot be obtained as a constant because the signal fades out with time as shown in Figures 5.1-5.2.

Figure 5.6 shows the spectrogram of the acoustic time series that is corrupted by the time-domain additive white Gaussian noise at an average SNR of 15 dB. The dispersion curve tracking at such an average SNR delivered by the SIR-PF, PBR-PF, and the new method are shown in Figures 5.7-5.9. The white dots represent positions of the estimated center frequency of modal waves that form the dispersion curves. At the beginning of the tracking, some modes merge and ambiguity in tracking is high. There are no significant differences between tracking results delivered by all filters at 500-650 ms. Figure 5.10 shows a comparison of spectrum estimation delivered by the three filters at time 485 ms. The leftmost, wide and non-symmetric mode is estimated

by two or more squared sinc functions. The rightmost mode, however, cannot be tracked by any filter. Also, most of the modal amplitudes estimated by each filter seem to not differ much from each other. At time 700-900 ms, the new method seems to overestimate the number of dispersion curves as shown in Figure 5.9. Figure 5.11, however, shows a proof of the superior performance of the new method as it can capture the 344-Hz and the 371-Hz frequency modes at time 715 ms, while the others cannot. Figures 5.12-5.14 show the probability mass function (PMF) of the number of modes delivered from the SIR-PF, PBR-PF, and the new method, respectively.

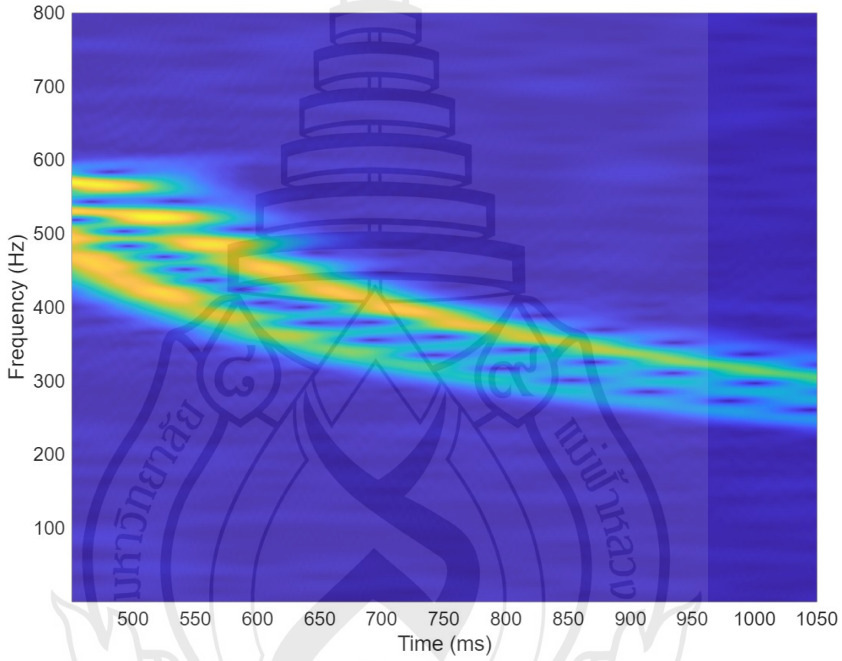


Figure 5.6 The noisy spectrogram at an average SNR of 15 dB

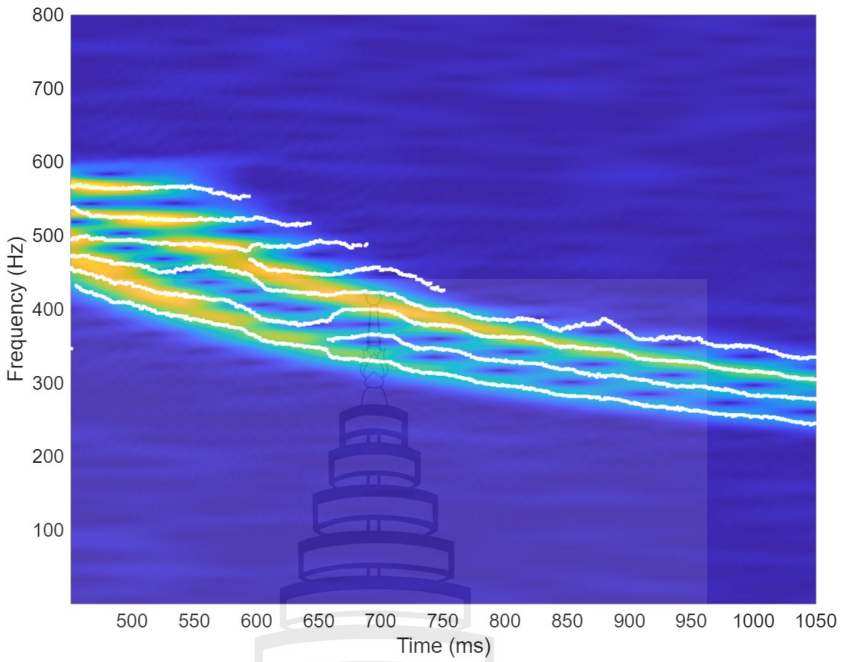


Figure 5.7 Dispersion curves at an average SNR of 15 dB as tracked by the SIR-PF

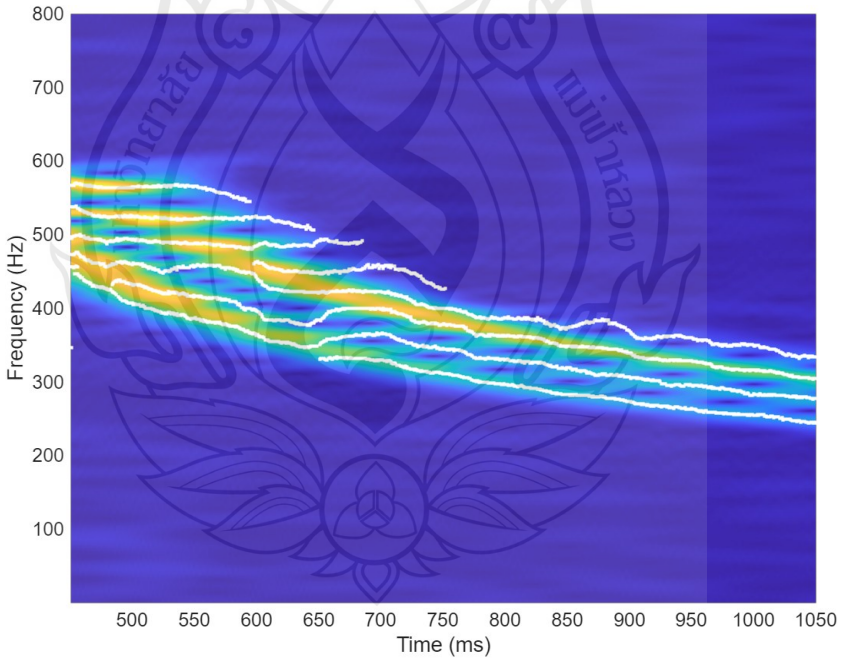


Figure 5.8 Dispersion curves at an average SNR of 15 dB as tracked by the PBR-PF

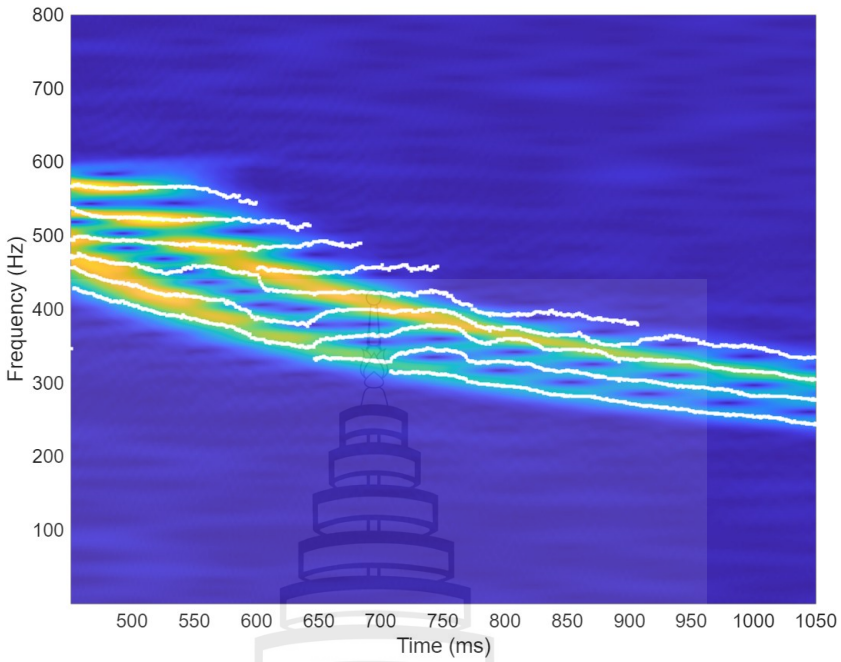


Figure 5.9 Dispersion curves at an average SNR of 15 dB as tracked by the proposed method

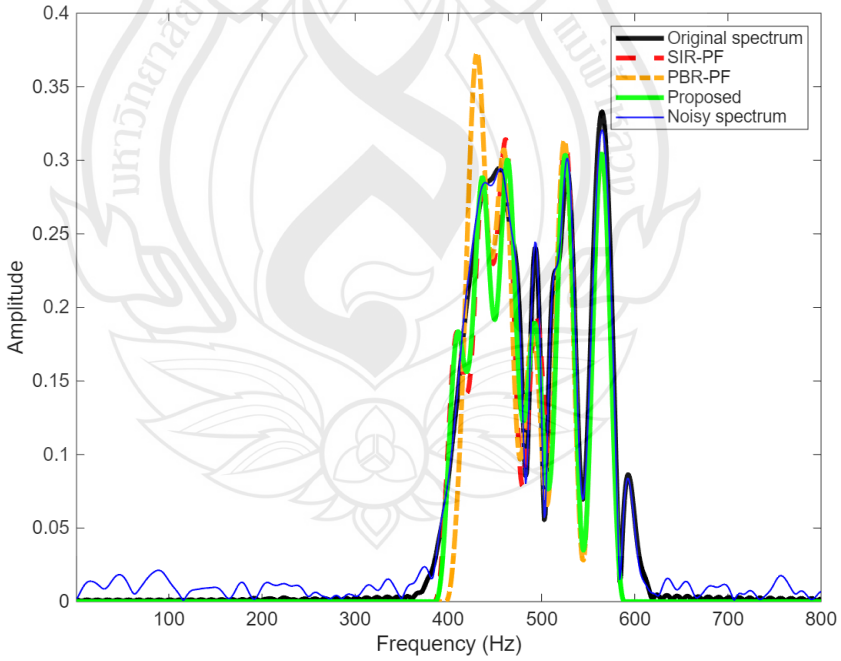


Figure 5.10 Spectrum estimation at time 485 ms for an average SNR of 15 dB

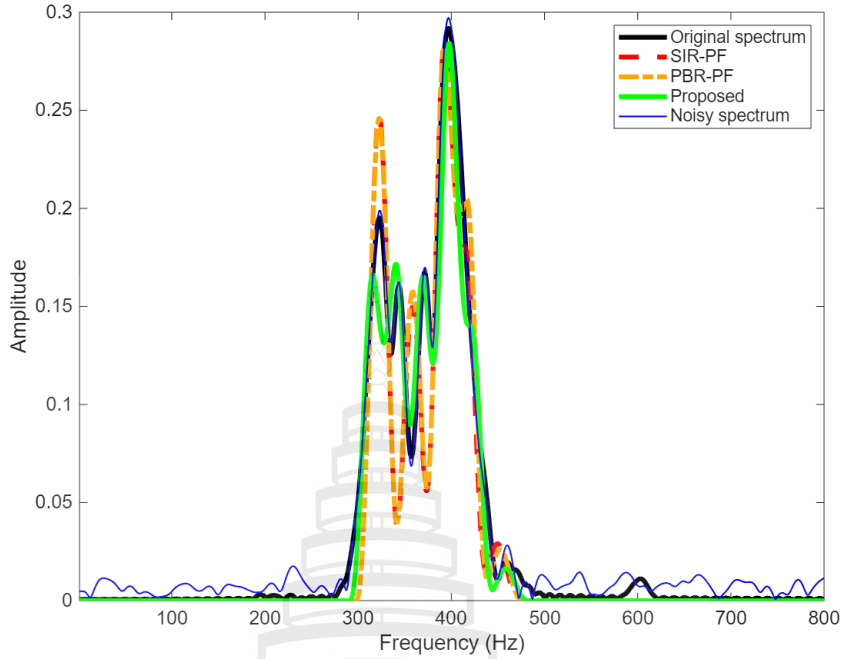


Figure 5.11 Spectrum estimation at time 715 ms for an average SNR of 15 dB

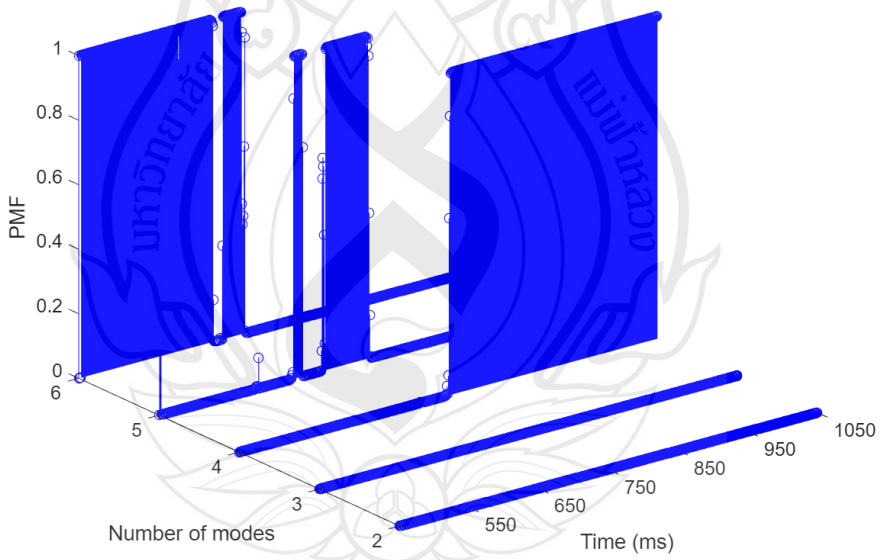


Figure 5.12 PMF of the number of modes for an average SNR of 15 dB delivered from the SIR-PF

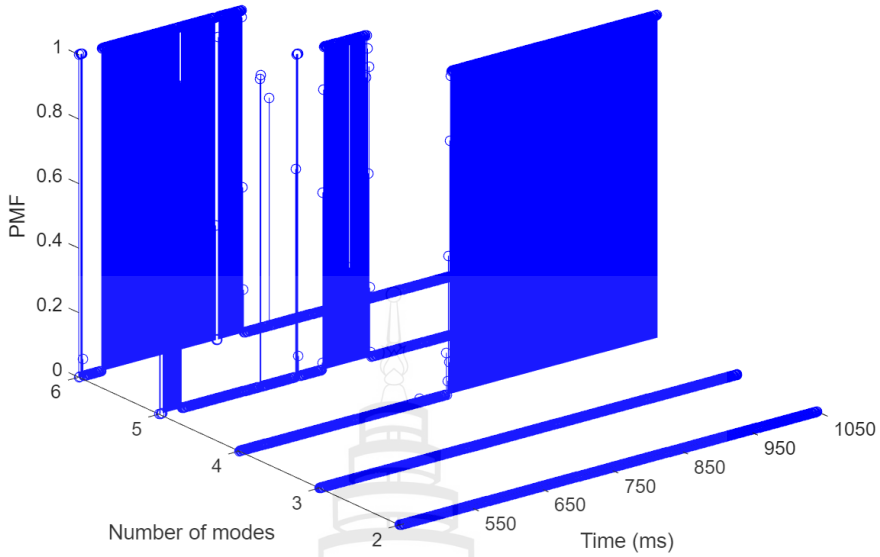


Figure 5.13 PMF of the number of modes for an average SNR of 15 dB delivered from the PBR-PF

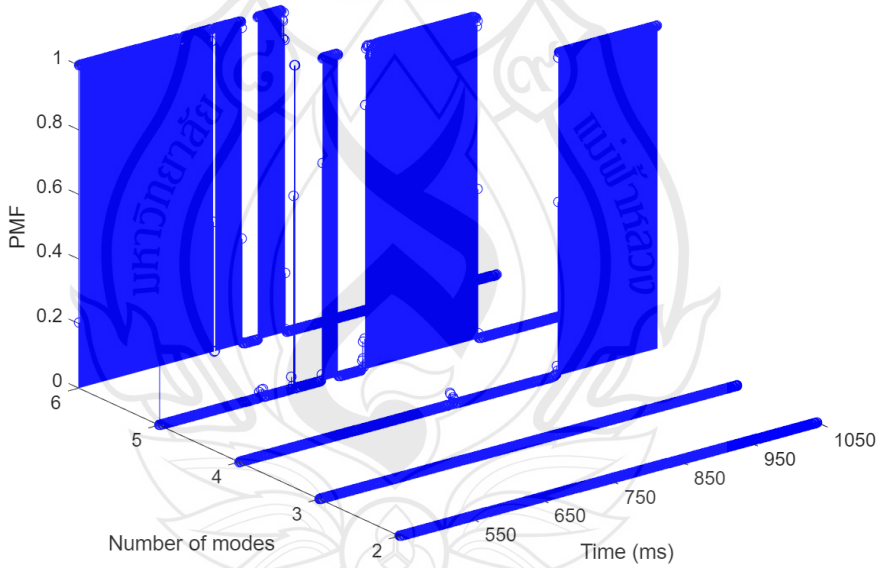


Figure 5.14 PMF of the number of modes for an average SNR of 15 dB delivered from the proposed method

Next, we show the results at an average SNR of 5 dB. The noisy spectrogram at such an average SNR is shown in Figure 5.15, while the dispersion curve tracking results are shown in Figures 5.16-5.18. For all filters, curves at the beginning of the spectrogram are not well tracked. The SIR-PF, delivers the poorest tracking at such time when compared to the PBR-PF and the new method. Both the SIR-PF and the

PBR-PF started to miss tracking the topmost mode at time 900 ms, as shown in Figures 5.16 and 5.17, respectively. The new method can capture such a mode; however, a false high-frequency mode shortly occurs between 900 and 950 ms as shown in Figure 5.18. This stems from the fact that GAs tries to find the new state vectors with higher likelihoods, while the likelihood value mainly affected by the closeness to the observation (i.e., noisy spectrum slices) as shown in Equation 5.30. Figure 5.19 shows the superior performance in spectrum slice estimation at time 950 ms of the new method. The shown stronger amplitude of false modes (i.e., observation noise) proves the side-effect of the new method, while only the new method can capture a 362-Hz frequency mode. Figures 5.20-5.22 show the probability mass function (PMF) of the number of modes delivered from the SIR-PF, PBR-PF, and the new method, respectively.

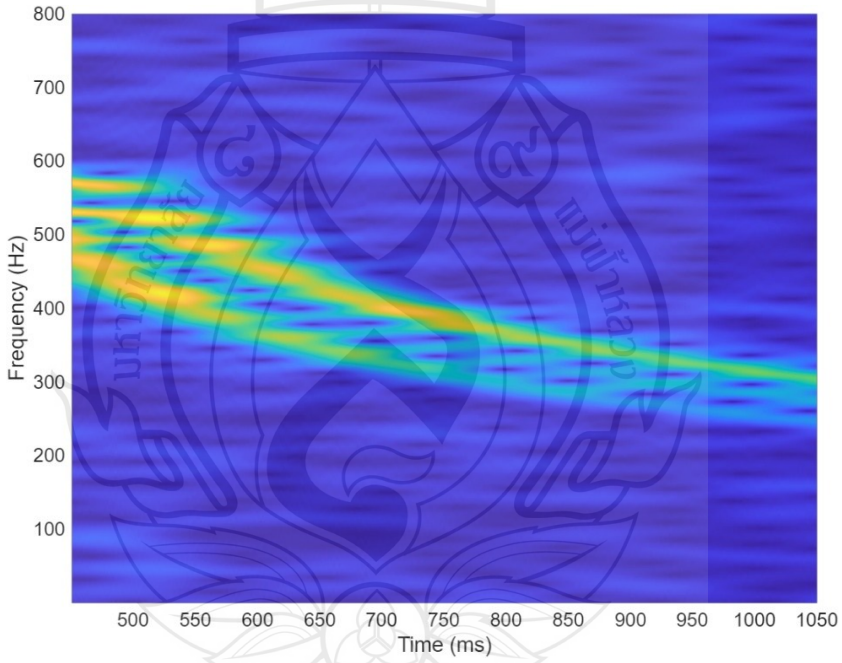


Figure 5.15 The noisy spectrogram at an average SNR of 5 dB

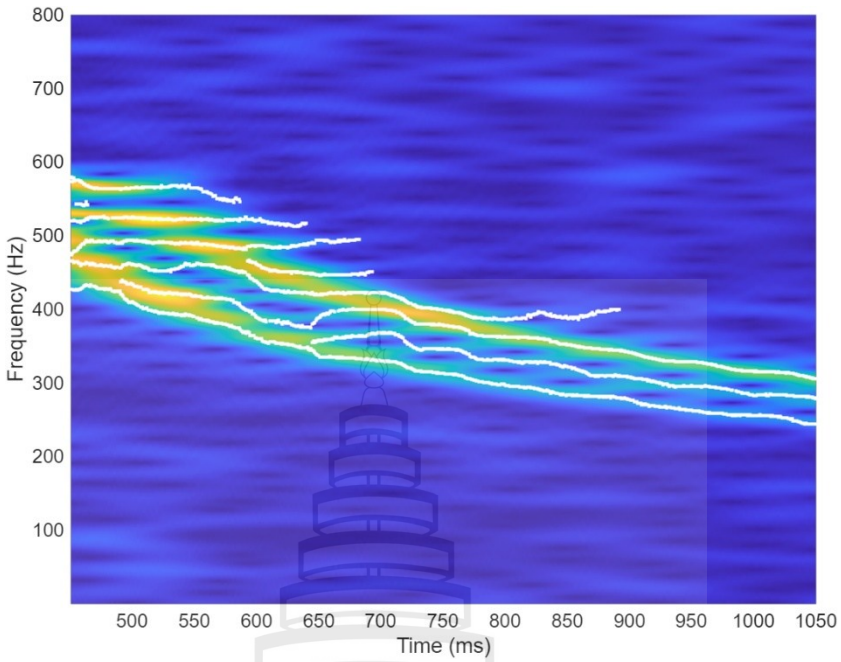


Figure 5.16 Dispersion curves at an average SNR of 5 dB as tracked by the SIR-PF

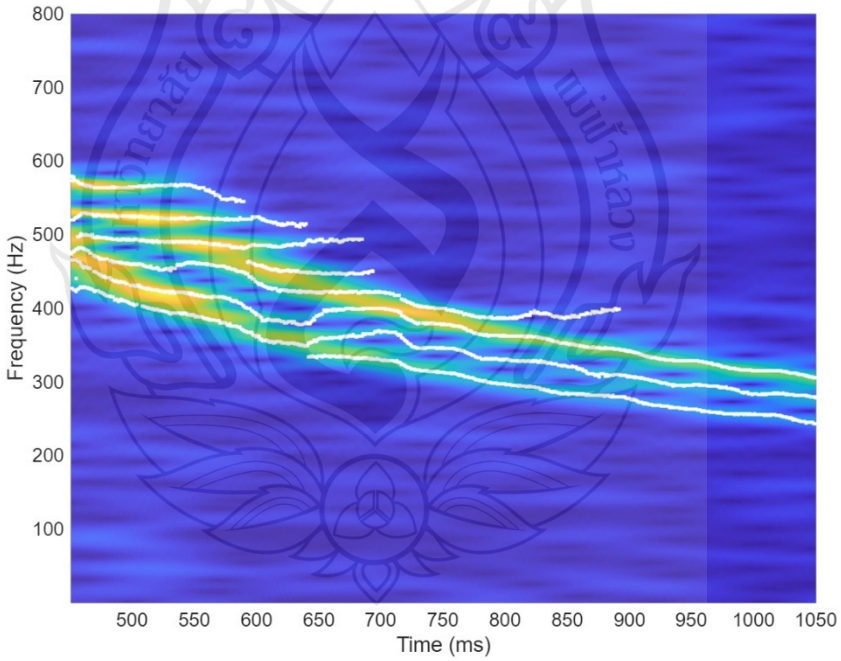


Figure 5.17 Dispersion curves at an average SNR of 5 dB as tracked by the PBR-PF

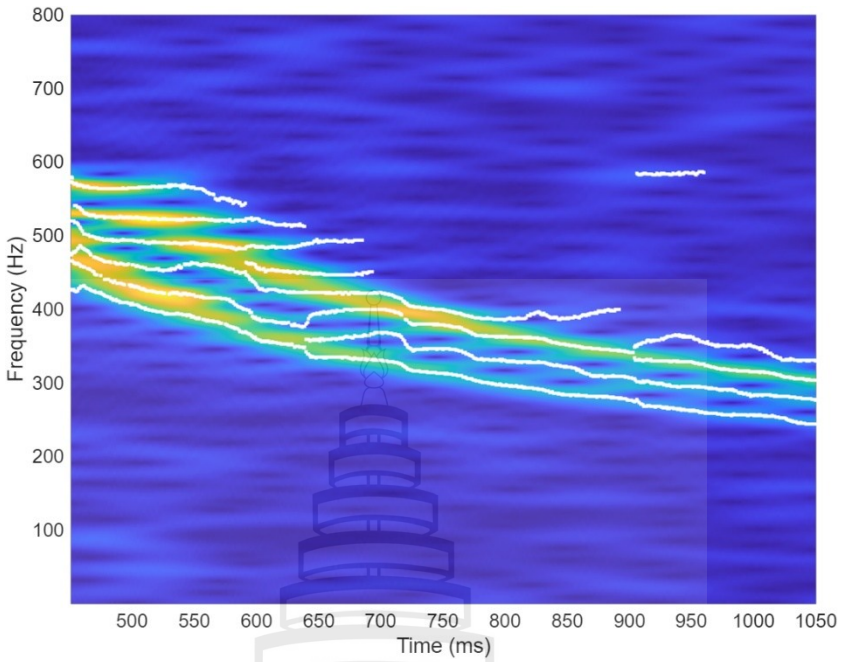


Figure 5.18 Dispersion curves at an average SNR of 5 dB as tracked by the proposed method

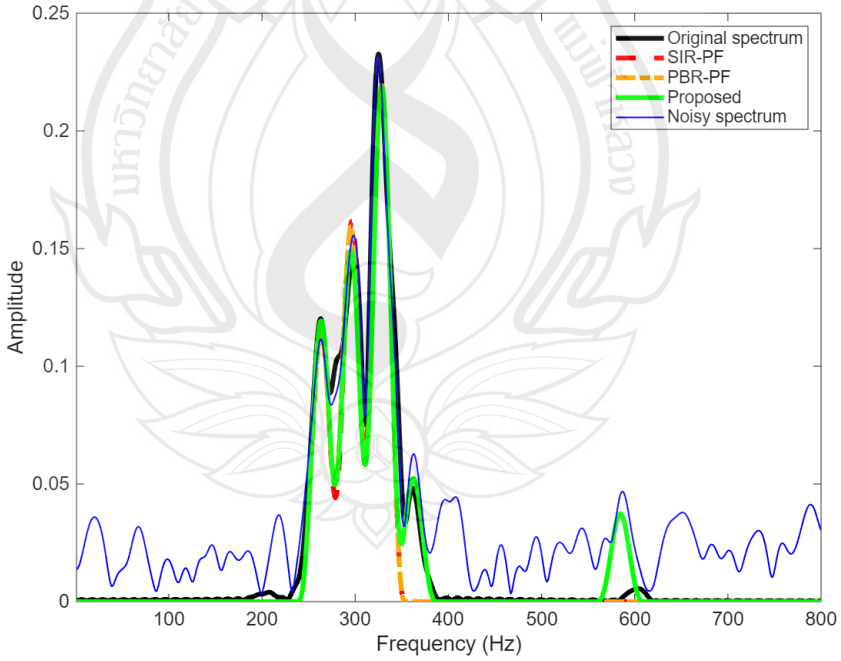


Figure 5.19 Spectrum estimation at time 950 ms for an average SNR of 5 dB

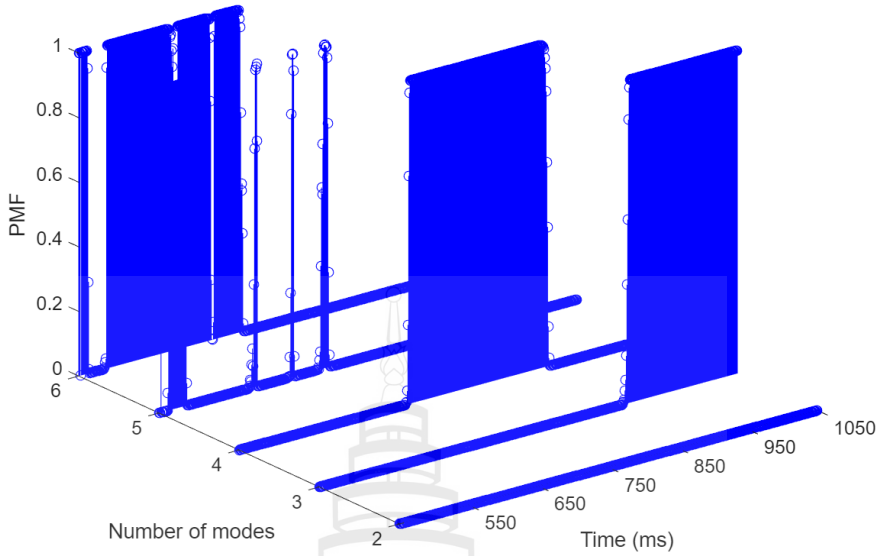


Figure 5.20 PMF of the number of modes for an average SNR of 5 dB delivered from the SIR-PF

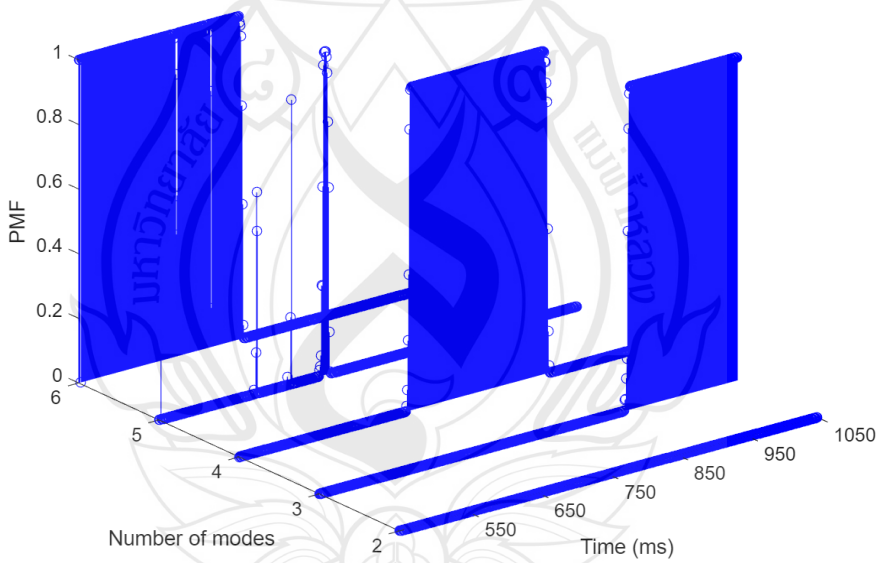


Figure 5.21 PMF of the number of modes for an average SNR of 5 dB delivered from the PBR-PF

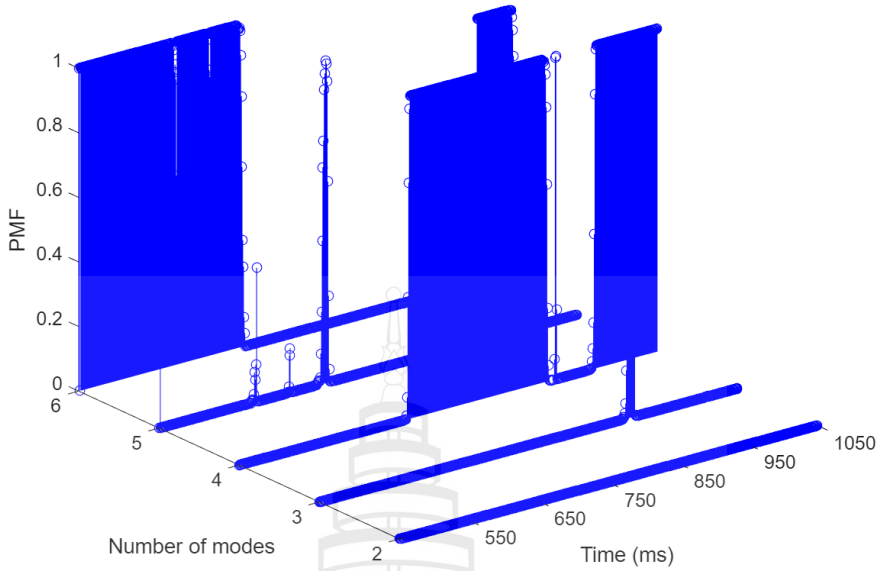


Figure 5.22 PMF of the number of modes for an average SNR of 5 dB delivered from the proposed method

Next, we perform the experiment when the average SNR is 0 dB where the noisy spectrogram at such an average SNR is shown in Figure 5.23. The dispersion curve tracking results are shown in Figures 5.24-5.26. The bottommost curve tracked by the SIR-PF has a discontinuity at around 900 ms, as shown in Figure 5.24. This discontinuity does not appear in the results delivered by the PBR-PF (in Figure 5.25) and the new method (in Figure 5.26). The dispersion curves at around 650 to 750 ms delivered by the new method (in Figure 5.26). look better than those delivered by the SIR-PF (in Figure 5.24). and the PBR-PF (in Figure 5.25) because the former has better continuity of the curves. However, the new method faces the most severe problem about delivering false modes as shown as short curves appearing on the TFR. This stems from the fact that the false modes can have higher amplitudes when the average SNR is low and they can be more likely to be misidentified as modal frequency. Figures 5.27-5.28 shows the exemplar comparison of spectrum estimation at time 729 ms and 927 ms, respectively. The new method is shown being able to capture all modes at such exemplar time steps, but the side-effect of misidentifying noise as false modes remains. Figures 5.29-5.31 show the probability mass function (PMF) of the number of modes delivered from the SIR-PF, PBR-PF, and the new method, respectively. Lower average SNRs cause the new method to prefer the big number of modes.

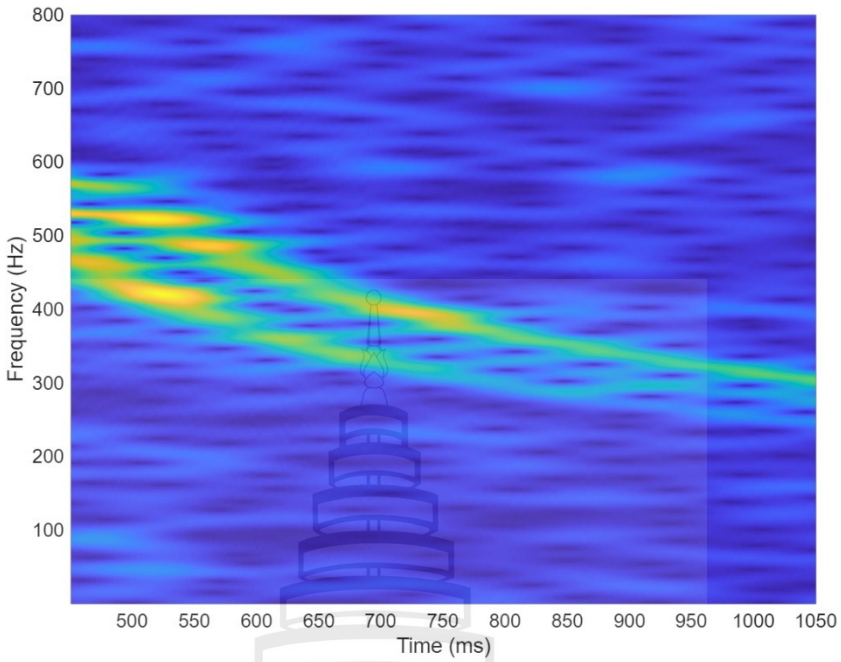


Figure 5.23 The extremely noisy spectrogram at an average SNR of 0 dB

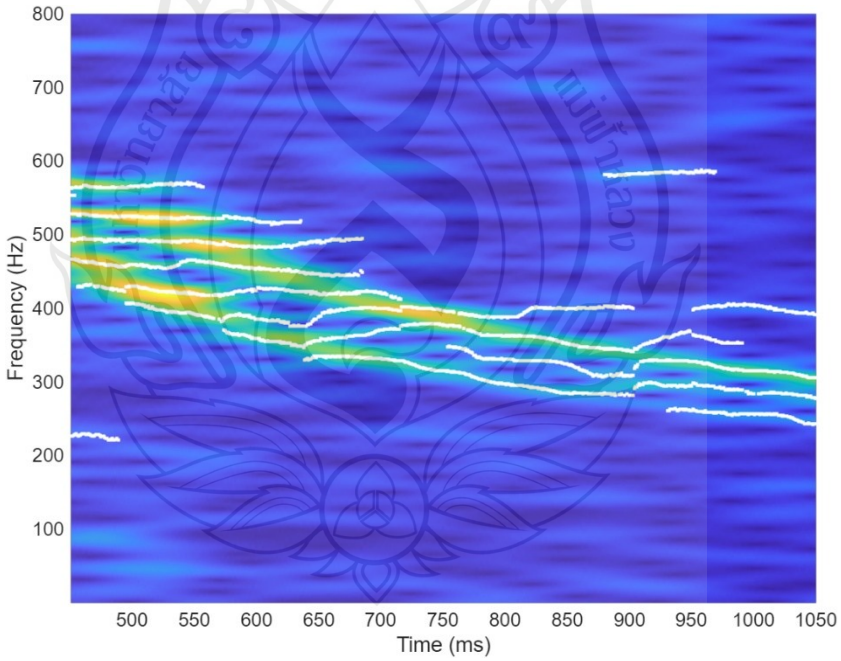


Figure 5.24 Dispersion curves at an average SNR of 0 dB as tracked by the SIR-PF

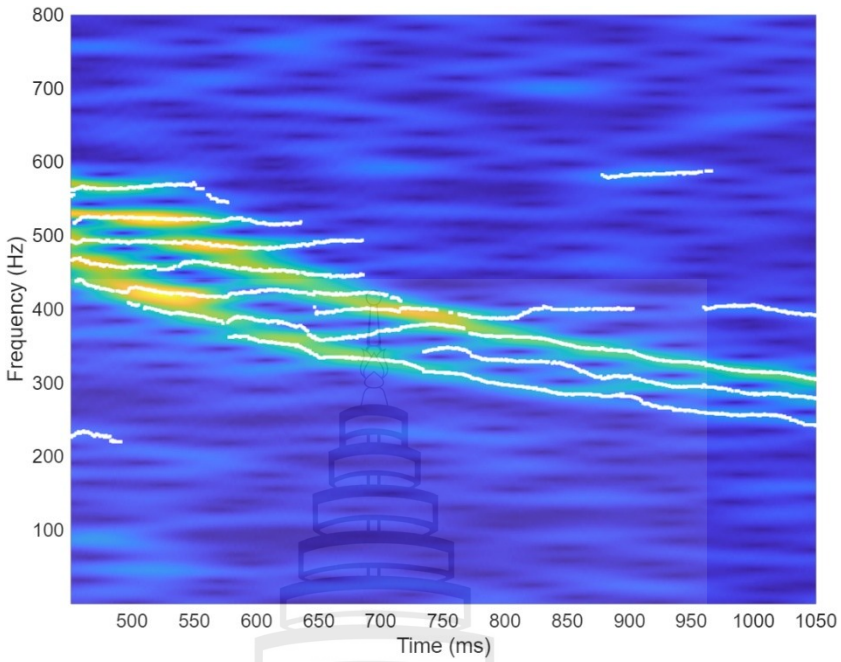


Figure 5.25 Dispersion curves at an average SNR of 0 dB as tracked by the PBR-PF

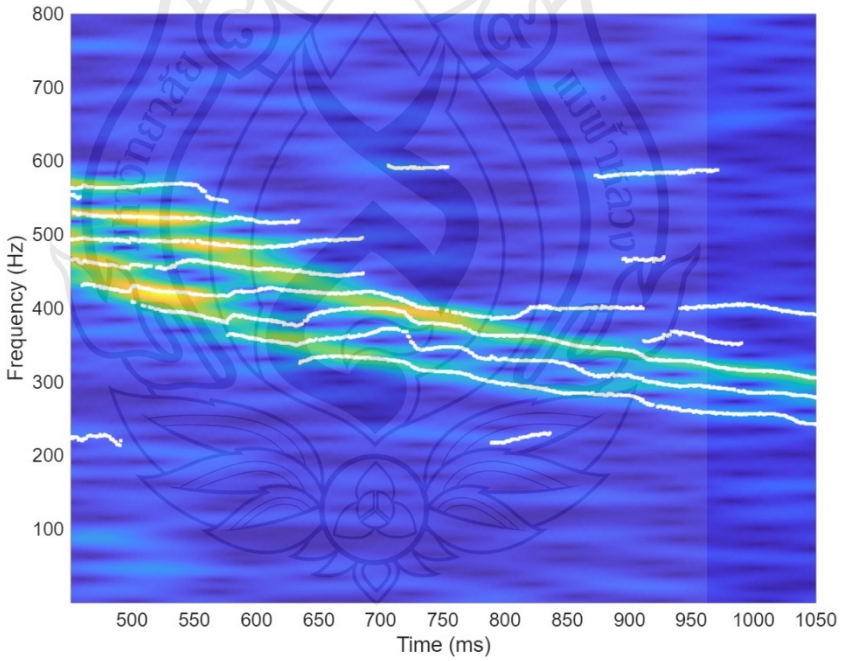


Figure 5.26 Dispersion curves at an average SNR of 0 dB as tracked by the proposed method

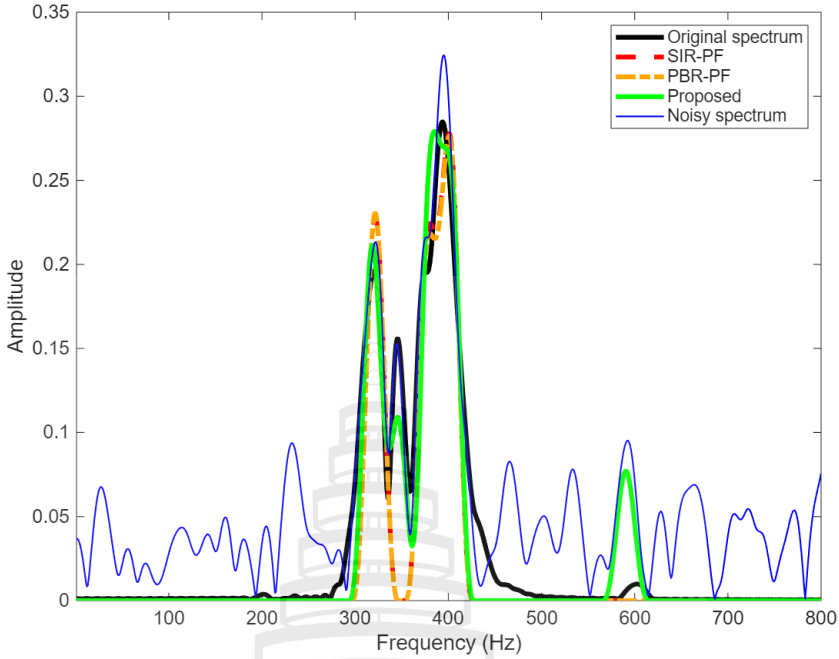


Figure 5.27 Spectrum estimation at time 729 ms for an average SNR of 0 dB

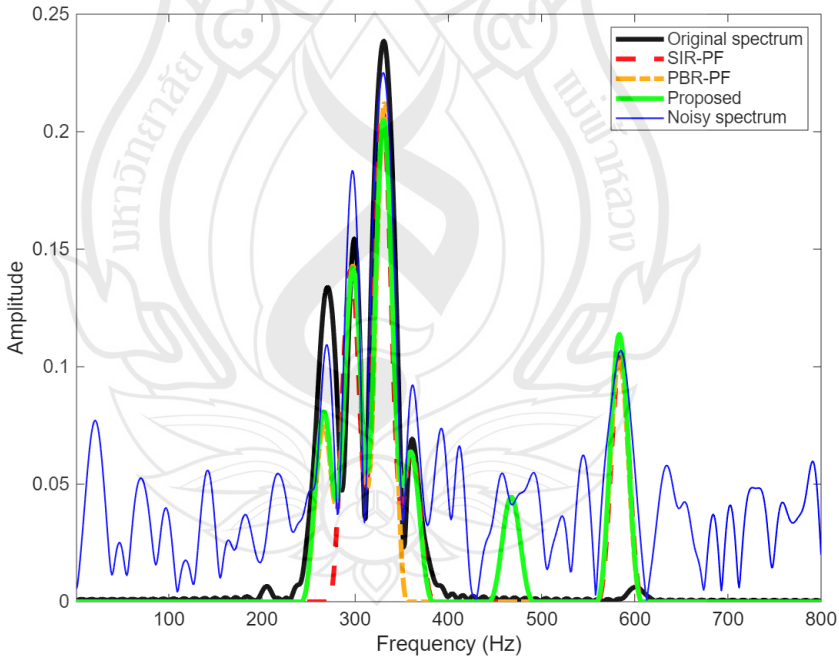


Figure 5.28 Spectrum estimation at time 927 ms for an average SNR of 0 dB

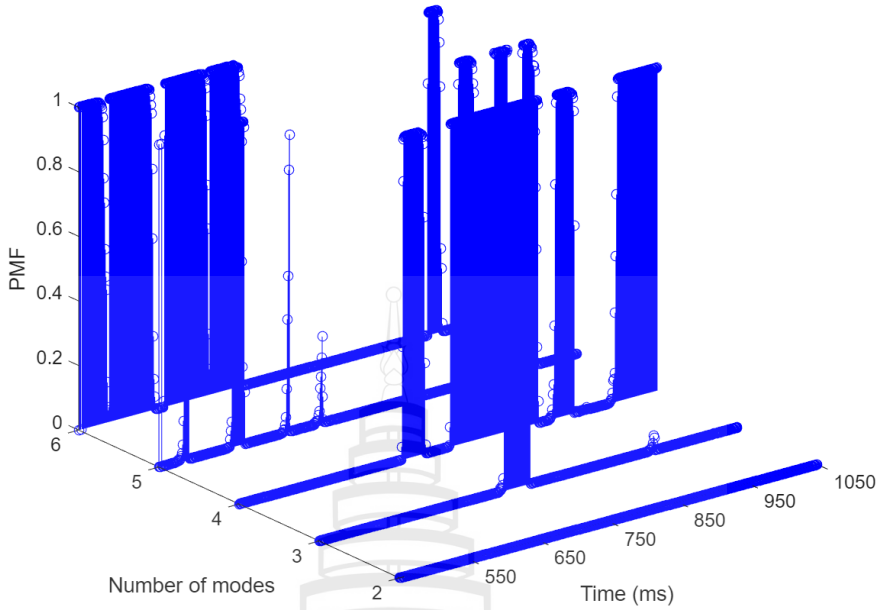


Figure 5.29 PMF of the number of modes for an average SNR of 0 dB delivered from the SIR-PF

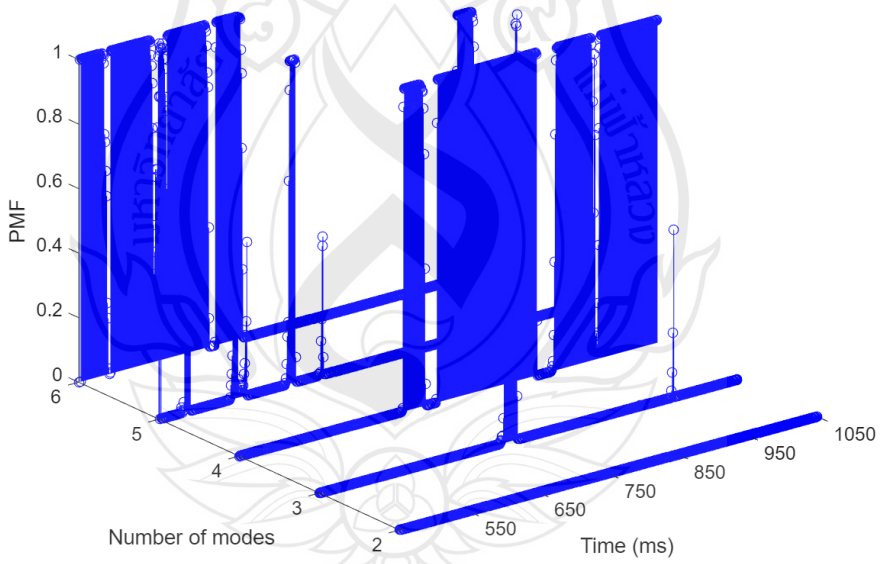


Figure 5.30 PMF of the number of modes for an average SNR of 0 dB delivered from the PBR-PF

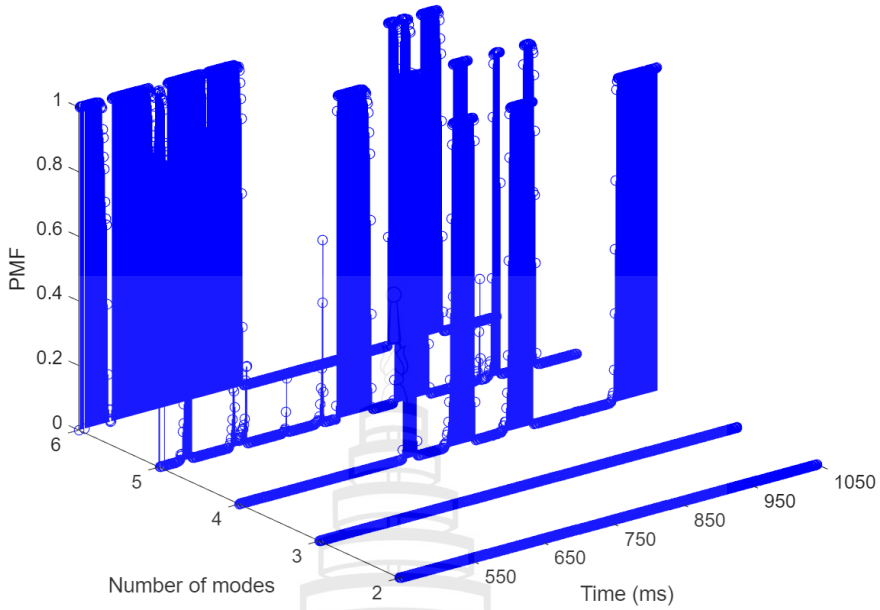


Figure 5.31 PMF of the number of modes for an average SNR of 0 dB delivered from the proposed method

Finally, we evaluate the performance in term of the root-mean-square errors which can be found as:

$$\text{Avg(RMSE)} = \frac{1}{R}\sum_{r=1}^{R}\sqrt{\frac{1}{(t_{last}-t_{first})+1}\sum_{k=t_{first}}^{t_{last}}||\mathbf{s}_k - \hat{\mathbf{s}}_{k,r}||^2}, \qquad (5.35)$$

where $\mathbf{s}_k$ denotes the true spectrum at time step k, while $\hat{\mathbf{s}}_{k,r}$ denotes the inferred spectrum at time step k of the r-th simulation run (found via Equation 5.34). Symbol $\|\cdot\|^2$ denotes the squared l^2 norm. Recall that t_{first} and t_{last} represent the first and the last time step selected according to Table 5.1 (Aunsri & Chamnongthai, 2021).

Table 5.2 presents a comparison of average RMSEs computed for each PF algorithm at different SNRs. At the average SNR of 15 dB. The new method yields the lowest RMSEs compared to those of the SIR-PF and the PBR-PF. However, as the average SNR decreases, the SIR-PF yields the lowest RMSEs compared to the PBR-PF and the new method. Recall that the PBR-PF prefers keeping and replicating high-weight particles without considering particle diversity. That is, the PBR-PF and the new method are sensitive to the high intensity noise where false modes are more likely to be misidentified. If the misidentified false modes can be eliminated by additional

techniques by the experts, the RMSEs delivered from the new method is supposed to be reduced.

Table 5.2 Average RMSEs

SNR	SIR-PF	PBR-PF	Proposed
15	0.6539	0.6511	**0.6342**
10	**0.6780**	0.6813	0.6881
5	**0.7440**	0.7528	0.7577
0	**0.8996**	0.9054	0.9127

CHAPTER 6

CONCLUSIONS

6.1 Conclusions

This dissertation presents a novel scheme for employing an adaptive GA efficiently in improving sequential state estimation performance under PF framework. To ensure diversity of new offspring particles, diversity of parents must be high. Recall that procedure of parent selection is similar to resampling but their objectives are different. High-weight particles that survive parent selection will be employed to create new offspring particles that belong to the same time step. In resampling, after low-weight particles are eliminated and high-weight particles are replicated, state values of copies of the latter will be updated via state evolution function in order to predict the true state at the next time step. While diversity of parents could be regained via roughening, particle degeneracy might return because some resampled particles might have lower weights after their state values were perturbed. Also, roughening could be employed only when size of state vectors was constant, according to the variance values found via Equation 2.37. Thus, instead of employing parent selection as done in traditional GAs and in GORPF, all of N weighted particles can be instantly employed as parents but they must first be classified as high-weight parents and low-weight parents.

Offspring state vectors calculation must be done for every pair of parents at every time step regardless of particle degeneracy measured as effective sample size (ESS) of the N parents. Recall that the maximum ESS denotes that each parent has the same weight, but it does not mean that this weight is actually high. In GORPF, offspring creation will be done only when ESS (found according to re-evaluated weights of the N post-roughening parents) is less than the preset threshold. According to simulation results in Chapter 4, the estimation performance delivered from GORPF was proved less reliable than those of our proposed method by having higher averages and variances of numerical errors, while our proposed method does not require as many preset GA

parameters as GORPF does. Also, averages and variances of computation time spent by GORPF were significantly higher than those of our proposed method.

Each low-weight parent must pair with a randomly selected high-weight parent in order to prevent any pair of two identical parents. Because number of low-weight parents and high-weight parents can be uneven, each high-weight parent may be repeatedly selected to pair with more than one low-weight parent. We set the probability of being selected of each high-weight parent to be uniform. The reasons are to ensure that: (1) the maximum-weight parent will not be preferred in order to ensure diversity of offspring particles, and (2) computation time can be saved because CDF of weights of high-weight parents is not required. Also, each created pair randomly selects only either flat crossover or modified Gaussian mutation (where mean values of the Gaussian PDF are state values of the high-weight parent) to find one new offspring particle in order to save computation time. That is, we treat state values of the high-weight parent as clues in finding the offspring state vector. Both GA operators are more efficient than blind perturbation done on state values of each particle. As presented in Chapter 4, performances of ASIR-PF and AFPF in estimating states from simulation state-space models were proved inferior to that of our proposed method by delivering higher RMSEs. Furthermore, GORPF selected each high-weight parent according to the CDF of weights of high-weight parents. This caused computation time of GORPF to be longer than those of IPF and our proposed method.

To ensure accuracy of state estimation, our proposed method accepts an offspring to replace its low-weight parent only if its weight is higher than that of its low-weight parent. As demonstrated in posterior PDFs reshaping in simulation one-dimensional state estimation (presented in Section 4.1), each low-weight parent in our proposed method could only either stay unchanged or randomly move to any new region where high-weight state values exist. Furthermore, our proposed method treats offspring particles whose weights are not lower than the weight threshold (which is found according to weights of all N parents and employed in parent classification) as additional high-weight parents. This scheme combats shortage of high-weight parents, especially in case of severe particle degeneracy. In IPF, Metropolis-Hastings (M-H) method was employed to randomly accept or reject the new offspring, while there were no any schemes for fixing shortage of high-weight parents. Average RMSEs of IPF

then were higher than those of our proposed method for simulation results in Chapter 4, while IPF and our proposed method delivered significant low variances of errors compared to those of the other state-of-the-art methods.

We also tested performance of the new method in estimating spectrum of an acoustics that disperses through an ocean waveguide in Chapter 5. GA operators were employed to find new offspring modal frequencies and new offspring modal amplitudes to ensure particle diversity. According to the results, stronger time-domain observation noise creates more false modes to be misidentified as the modal frequency. In other words, false dispersion curves are more likely to appear in the tracking results delivered by the new method. Although the new method showed superior performance in capturing the modal frequencies, the misidentification of false modes seems to be a side-effect of employing the new method. Consequently, the RMSEs of the new method are higher than those delivered from the SIR-PF and the PBR-PF for such low SNRs. The RMSEs should be reduced if the problem of the false mode misidentification can be solved.

6.2 Limitations

Adaptive GAs can be employed only when the condition $ESS_k < N$ is satisfied. When $ESS_k = N$ or weights of all N parents are same (but not necessarily high), parent classification will be impossible. In practice, it is difficult to achieve such maximum ESS, especially when size of state vector is high or number of particles N is sufficient. However, bigger state vectors decrease probability of finding the offspring whose weight is higher than that of its low-weight parent. Such curse of dimensionality also leads to spending more computation time in finding new offspring state values. This is an unavoidable tradeoff between state estimation performance and computation time. As previously shown in Chapter 4, SIR-PF spent the shortest computation time for both one-dimensional state and multidimensional state estimation, while RMSEs delivered from SIR-PF were higher than those of our proposed method.

Classifying parents as high-weight parents and low-weight parents can prevent any pair of two identical parents. However, in case the parents are state vectors, it can

be possible that m-th vector component of the low-weight parent and m-th vector component of the high-weight parent have same state value. If such case happens, the new offspring state value found using flat crossover for that m-th vector component, according to Equation 2.43, will be same to those of the two parents. If all state values in a vector have different units, we cannot swap the order of the state values and diversity of state values of that m-th vector component can be low.

Shortage of high-weight parents in case particle degeneracy is severe could be mitigated by adding new offspring particles whose weights were not less than the preset threshold as new members of set of high-weight parents. However, all pairs of parents cannot be created in advance because each pair must create an offspring sequentially and the newly added high-weight parent can be available to be selected to form the next pair of parents.

The new method prevents the parent particles from being replaced by the low-quality offspring particles. However, there are no schemes in GAs for ensuring that the weight of the new offspring candidate will always be higher than the weight of its low-weight parent. At any time step, there can be chances that the post-GA particle swarm will be the same as the original population because every new offspring has its weight which is lower than the weight of their respective low-weight parent; all of these offspring particles are then rejected.

6.3 Future Work

State values of any two parent particles can affect search space of the to-be-drawn offspring state values. If difference between the two parent state values is high, flat crossover should be preferred because search space is sufficiently large. On the contrary, if difference between the two parent state values is low, modified Gaussian mutation can be employed to find new state values that are located outside the small search space. That is, size of difference between the two parent state values should also be considered in randomly choosing a GA operator, not just ESS found according to weights of the N parents. Furthermore, bounds of the population (i.e., minimum and maximum state values of the particle swarm) should also be taken into consideration.

New schemes should be proposed and incorporated with GAs to ensure that the new offspring will be better than its low-weight parent in order to save computation time for offspring weight checking.

According to the results in Chapter 5, the new method could be employed under the MMPF framework where size of each particle (i.e., state vector) is uneven. This pilot study can then be extended to the employment on more complicated systems and applications.

REFERENCES

Ahwiadi, M., & Wang, W. (2020). An adaptive particle filter technique for system state estimation and prognosis. *IEEE Transactions on Instrumentation and Measurement*, *69*(9), 6756-6765. https://doi.org/10.1109/TIM.2020.2973850

Alam, T., Qamar, S., Dixit, A., & Benaida, M. (2020). Genetic algorithms: Reviews, implementations and applications. *International Journal of Engineering Pedagogy*, *10*(6), 57-77. https://doi.org/10.3991/ijep.v10i6.14567

Andrieu, C., Davy, M., & Doucet, A. (2003). Efficient particle filtering for jump Markov systems. Application to time-varying autoregressions. *IEEE Transactions on Signal Processing*, *51*(7), 1762-1770. https://doi.org/10.1109/TSP.2003.810284

Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, *50*(2), 174-188. https://doi.org/10.1109/78.978374

Aunsri, N. (2018a). Effects of window functions on the sequential Bayesian filtering based frequency estimation. In *21st International Symposium on Wireless Personal Multimedia Communications (WPMC)* (pp. 411-415). IEEE. https://doi.org/10.1109/WPMC.2018.8713162

Aunsri, N. (2018b). Seismic events estimation under noisy environments using multiple model particle filter. In *15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)* (pp. 793-797). IEEE. https://doi.org/10.1109/ECTICon.2018.8620047

Aunsri, N. (2019). Sequential Bayesian filtering with particle smoother for improving frequency estimation in frequency domain approach. In *22nd International Symposium on Wireless Personal Multimedia Communications (WPMC)* (pp. 1-5). IEEE. https://doi.org/10.1109/WPMC48795.2019.9096101

Aunsri, N., & Chamnongthai, K. (2019). Particle filtering with adaptive resampling scheme for modal frequency identification and dispersion curves estimation in ocean acoustics. *Applied Acoustics*, *154*, 90-98. https://doi.org/10.1016/j.apacoust.2019.04.018

Aunsri, N., & Chamnongthai, K. (2021). Stochastic description and evaluation of ocean acoustics time-series for frequency and dispersion estimation using particle filtering approach. *Applied Acoustics*, *178*, Article 108010. https://doi.org/10.1016/j.apacoust.2021.108010

Aunsri, N., & Michalopoulou, Z.-H. (2014). Sequential filtering for dispersion tracking and sediment sound speed inversion. *The Journal of the Acoustical Society of America*, *136*(5), 2665-2674. https://doi.org/10.1121/1.4897400

Aunsri, N., Pipatphol, K., Thikeaw, B., Robroo, S., & Chamnongthai, K. (2021). A novel adaptive resampling scheme for sequential Bayesian filtering to improve frequency estimation of time-varying signals. *Heliyon, 7*(4), Article e06768. https://doi.org/10.1016/j.heliyon.2021.e06768

Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette (Ed.), *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms* (pp. 14-21). Psychology Press. https://doi.org/10.4324/9780203761595

Bhat, P. G., Subudhi, B. N., Veerakumar, T., Di Caterina, G., & Soraghan, J. J. (2021). Target tracking using a mean-shift occlusion aware particle filter. *IEEE Sensors Journal*, *21*(8), 10112-10121. https://doi.org/10.1109/JSEN.2021.3054815

Boashash, B. (2016). Heuristic formulation of time-frequency distributions. In B. Boashash (Ed.), *Time-frequency signal analysis and processing*: *A comprehensive reference* (2nd ed., pp. 65-102). Academic Press.

Candy, J. V. (2016). *Bayesian signal processing: Classical, modern and particle filtering methods* (2nd ed.). John Wiley & Sons. https://doi.org/10.1002/9781119125495

Cappé, O., Godsill, S. J., & Moulines, E. (2007). An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, *95*(5), 899-924. https://doi.org/10.1109/JPROC.2007.893250

Carpenter, J., Clifford, P., & Fearnhead, P. (1999). Improved particle filter for nonlinear problems. *IEE Proceedings,* (*Radar and Signal Processing*), *146*(1), 2-7. https://doi.org/10.1049/ip-rsn:19990255

De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems* [Doctoral dissertation, University of Michigan]. Deep Blue Documents. https://deepblue.lib.umich.edu/items/4f34b899-06d6-456e-a69e-43fea0d8b43b

Doucet, A., Godsill, S., & Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, *10*(3), 197-208. https://doi.org/10.1023/A:1008935410038

Drachal, K., & Pawłowski, M. (2021). A review of the applications of genetic algorithms to forecasting prices of commodities. *Economies*, *9*(1), Article 6. https://doi.org/10.3390/economies9010006

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, *99*(C5), 10143-10162. https://doi.org/10.1029/94JC00572

Garzelli, A., Capobianco, L., & Nencini, F. (2008). Fusion of multispectral and panchromatic images as an optimisation problem. In T. Stathaki (Ed.), *Image fusion: Algorithms and applications* (pp. 223-250). Academic Press.

Gordon, N. J., Salmond, D. J., & Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F* (*Radar and Signal Processing*), *140*(2), 107-113. https://doi.org/10.1049/ip-f-2.1993.0015

Han, H., Ding, Y.-S., Hao, K.-R., & Liang, X. (2011). An evolutionary particle filter with the immune genetic algorithm for intelligent video target tracking. *Computers and Mathematics with Applications*, *62*(7), 2685-2695. https://doi.org/10.1016/j.camwa.2011.06.050

Han, X., Lin, H., Li, Y., Ma, H., & Zhao, X. (2015). Adaptive fission particle filter for seismic random noise attenuation. *IEEE Geoscience and Remote Sensing Letters*, *12*(9), 1918-1922. https://doi.org/10.1109/LGRS.2015.2438229

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, *57*(1), 97-109. https://doi.org/10.2307/2334940

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66-73. http://doi.org/10.1038/scientificamerican0792-66

Huillery, J., Millioz, F., & Martin, N. (2008). On the description of spectrogram probabilities with a chi-squared law. *IEEE Transactions on Signal Processing*, *56*(6), 2249-2258. https://doi.org/10.1109/TSP.2007.916125

Julier, S. J., & Uhlmann, J. K. (1997). A new extension of the Kalman filter to nonlinear systems. In I. Kadar (Ed.), *Signal Processing, Sensor Fusion, and Target Recognition VI* (pp. 182-193). SPIE. https://doi.org/10.1117/12.280797

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*(1), 35-45. https://doi.org/10.1115/1.3662552

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: Past, present, and future. *Multimedia Tools and Applications*, *80*(5), 8091-8126. https://doi.org/10.1007/s11042-020-10139-6

Katzfuss, M., Stroud, J. R., & Wikle, C. K. (2016). Understanding the ensemble Kalman filter. *The American Statistician*, *70*(4), 350-357. https://doi.org/10.1080/00031305.2016.1141709

Krumm, J. (2010). Processing sequential sensor data. In J. Krumm (Ed.), *Ubiquitous computing fundamentals* (pp. 353-380). Chapman & Hall/CRC. https://doi.org/10.1201/9781420093612

Kuptametee, C., & Aunsri, N. (2022a). A review of resampling techniques in particle filtering framework. *Measurement*, *193*, Article 110836. https://doi.org/10.1016/j.measurement.2022.110836

Kuptametee, C., & Aunsri, N. (2022b). Particle filtering with adaptive diversifying scheme for abruptly changing hidden states estimation. In *6th International Conference on Information Technology (InCIT)* (pp. 358-362). IEEE. https://doi.org/10.1109/InCIT56086.2022.10067827

Kuptametee, C., & Aunsri, N. (2022c). Sequential frequency estimation using auxiliary particle filter. In *6th International Conference on Information Technology (InCIT)* (pp. 363-367). IEEE. https://doi.org/10.1109/InCIT56086.2022.10067382

Kuptametee, C., & Aunsri, N. (2023). Intelligent genetic crossover algorithm for improving state estimation in particle filtering. In *7th International Conference on Information Technology (InCIT)* (pp. 550-555). IEEE. https://doi.org/10.1109/InCIT60207.2023.10413189

Kuptametee, C., Michalopoulou, Z.-H., & Aunsri, N. (2024). A review of efficient applications of genetic algorithms to improve particle filtering optimization problems. *Measurement*, *224*, Article 113952. https://doi.org/10.1016/j.measurement.2023.113952

Larose, D. T. (2006). Genetic algorithms. *Data mining methods and models* (pp. 240-264). John Wiley & Sons. https://doi.org/10.1002/0471756482

Lewis, J. & Chase, J. (2014). *Java software structures: Designing and using data structures* (4th international ed.). Pearson Education.

Li, T., Bolić, M., & Djurić, P. M. (2015). Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, *32*(3), 70-86. https://doi.org/10.1109/MSP.2014.2330626

Li, T., Sattar, T. P., & Tang, D. (2013). A fast resampling scheme for particle filters. In *2013 Constantinides International Workshop on Signal Processing (CIWSP)* (pp. 1-4). IEEE. https://doi.org/10.1049/ic.2013.0002

Martino, L., Elvira, V., & Louzada, F. (2017). Effective sample size for importance sampling based on discrepancy measures. *Signal Processing*, *131*, 386-401.

Maybeck, P. S. (1982). *Stochastic models, estimation and control* (Vol. 2). Academic Press.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs* (3rd ed.). Springer-Verlag. https://doi.org/10.1007/978-3-662-03315-9

Michalopoulou, Z.-H., & Aunsri, N. (2018). Environmental inversion using dispersion tracking in a shallow environment. *The Journal of the Acoustical Society of America*, *143*(3), EL188-EL193. https://doi.org/10.1121/1.5026245

Musso, C., Oudjane N., & Le Gland, F. (2001). Improving regularised particle filters. In A. Doucet, N. de Freitas, & N. Gordon (Eds.), *Sequential Monte Carlo methods in practice* (pp. 247-272). Springer. https://doi.org/10.1007/978-1-4757-3437-9

Park, S., Hwang, J. P., Kim, E., & Kang, H.-J. (2009). A new evolutionary particle filter for the prevention of sample impoverishment. *IEEE Transactions on Evolutionary Computation*, *13*(4), 801-809. https://doi.org/10.1109/TEVC.2008.2011729

Pitt, M. K., & Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, *94*(446), 590-599. https://doi.org/10.2307/2670179

Radcliffe, N. J. (1990). *Genetic neural networks on MIMD computers* [Doctoral dissertation, University of Edinburgh]. Edinburgh Research Archive. https://era.ed.ac.uk/handle/1842/11288

Ristic, B., Arulampalam, M. S., & Gordon, N. (2004). *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House.

Roonizi, A. K. (2022). Kalman filtering in non-Gaussian model errors: A new perspective [Tips & Tricks], *IEEE Signal Processing Magazine*, *39*(3), 105-114. https://doi.org/10.1109/MSP.2021.3134635

Saenmuang, S., & Aunsri, N. (2019). A new spinach respiratory prediction method using particle filtering approach. *IEEE Access*, *7*, 131559-131566. https://doi.org/10.1109/ACCESS.2019.2941176

Tan, L., & Jiang, J. (2019). *Digital signal processing: Fundamental and applications* (3rd ed.). Academic Press. https://doi.org/10.1016/C2017-0-02319-4

van Leeuwen, P. J. (2020). A consistent interpretation of the stochastic version of the ensemble Kalman filter. *Quarterly Journal of the Royal Meteorological Society*, *146*(731), 2815-2825. https://doi.org/10.1002/qj.3819

Wan, E. A., & van der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. In *IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium* (pp. 153-158). IEEE. https://doi.org/10.1109/ASSPCC.2000.882463

Wang, Y., Wang, X., Shan, Y., & Cui, N. (2020). Quantized genetic resampling particle filtering for vision-based ground moving target tracking. *Aerospace Science and Technology*, *103*, Article 105925. https://doi.org/10.1016/j.ast.2020.105925

Yang, T. C. (1984). A method for measuring the frequency dispersion for broadband pulses propagated to long ranges. *The Journal of the Acoustical Society of America*, *76*(1), 253-261. https://doi.org/10.1121/1.391102

Yardim, C., Michalopoulou, Z.-H., & Gerstoft, P. (2011). An overview of sequential Bayesian filtering in ocean acoustics. *IEEE Journal of Oceanic Engineering*, *36*(1), 71-89. https://doi.org/10.1109/JOE.2010.2098810

Yin, S., & Zhu, X. (2015). Intelligent particle filter and its application to fault detection of nonlinear system. *IEEE Transactions on Industrial Electronics*, *62*(6), 3852-3861. https://doi.org/10.1109/TIE.2015.2399396

Yin, S., Zhu, X., Qiu, J., & Gao, H. (2016). State estimation in nonlinear system using sequential evolutionary filter. *IEEE Transactions on Industrial Electronics*, *63*(6), 3786-3794. https://doi.org/10.1109/TIE.2016.2522382

Yu, M., Li, H., Jiang, W., Wang, H., & Jiang, C. (2019). Fault diagnosis and RUL prediction of nonlinear mechatronic system via adaptive genetic algorithm-particle filter. *IEEE Access*, *7*, 11140-11151. https://doi.org/10.1109/ACCESS.2019.2891854

Zafar, T., Mairaj, T., Alam, A., & Rasheed, H. (2020). Hybrid resampling scheme for particle filter-based inversion. *IET Science, Measurement and Technology*, *14*(4), 396-406. https://doi.org/10.1049/iet-smt.2018.5531

Zhang, X., Liu, D., Yang, Y., & Liang, J. (2021). An intelligent particle filter with adaptive M-H resampling for liquid-level estimation during silicon crystal growth. *IEEE Transactions on Instrumentation and Measurement*, *70*, Article 3502812. https://doi.org/10.1109/TIM.2020.3026760

Zhou, N., Lau, L., Bai, R., & Moore, T. (2021). A genetic optimization resampling based particle filtering algorithm for indoor target tracking. *Remote Sensing*, *13*(1), Article 132. https://doi.org/10.3390/rs13010132

Zhou, W., Liu, L., & Hou, J. (2019). Firefly algorithm-based particle filter for nonlinear systems. *Circuits, Systems, and Signal Processing*, *38*(4), 1583-1595. https://doi.org/10.1007/s00034-018-0927-0

Zorych, I., & Michalopoulou, Z.-H. (2008). Particle filtering for dispersion curve tracking in ocean acoustics. *The Journal of the Acoustical Society of America*, *124*(2), EL45-EL50. https://doi.org/10.1121/1.2947628

CURRICULUM VITAE

NAME Chanin Kuptametee

EDUCATIONAL BACKGROUND

2019 Bachelor of Engineering
Information and Communication Engineering
Mae Fah Luang University

SCHOLARSHIP

2022 Mae Fah Luang University Dissertation Support Grant

2020 Mae Fah Luang University Graduate Scholarship

PUBLICATION

ARTICLES

Kuptametee, C., Michalopoulou, Z.-H., & Aunsri, N. (2025). *Adaptive genetic algorithm for modal frequency and dispersion curve estimation in particle filtering framework* [Manuscript in preparation]. School of Applied Digital Technology, Mae Fah Luang University.

Kuptametee, C., Michalopoulou, Z.-H., & Aunsri, N. (2024). A review of efficient applications of genetic algorithms to improve particle filtering optimization problems. *Measurement*, *224*, Article 113952. https://doi.org/10.1016/j.measurement.2023.113952

Kuptametee, C., Michalopoulou, Z.-H., & Aunsri, N. (2023). Adaptive genetic algorithm-based particle herding scheme for mitigating particle impoverishment. *Measurement*, *214*, Article 112785. https://doi.org/10.1016/j.measurement.2023.112785

Kuptametee, C., & Aunsri, N. (2022). A review of resampling techniques in particle filtering framework. *Measurement*, *193*, Article 110836. https://doi.org/10.1016/j.measurement.2022.110836

PUBLICATION (continued)

PRESENTATIONS

Kuptametee, C., & Aunsri, N. (2023). Intelligent genetic crossover algorithm for improving state estimation in particle filtering. In *7th International Conference on Information Technology (InCIT)* (pp. 550-555). IEEE. Thailand. https://doi.org/10.1109/InCIT60207.2023.10413189 (Best paper award)

Khine, M. H. H., Kuptametee, C., & Aunsri, N. (2023). Improving reliability of state estimation by particle filter with Cauchy likelihood function. In *7th International Conference on Information Technology (InCIT)* (pp. 556-561), Chiang Rai, Thailand. https://doi.org/10.1109/InCIT60207.2023.10413186

Kuptametee, C., & Aunsri, N. (2023). Sequential abruptly changing hidden states estimation using adaptive particle impoverishment mitigation scheme. In *2023 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)* (pp. 302-307). IEEE. https://doi.org/10.1109/ECTIDAMTNCON57770.2023.10139728

Kuptametee, C., & Aunsri, N. (2022). Particle filtering with adaptive diversifying scheme for abruptly changing hidden states estimation. In *6th International Conference on Information Technology (InCIT)* (pp. 358-362). IEEE. https://doi.org/10.1109/InCIT56086.2022.10067827 (Best paper award)

Kuptametee, C., & Aunsri, N. (2022). Sequential frequency estimation using auxiliary particle filter. In *6th International Conference on Information Technology (InCIT)* (pp. 363-367). IEEE. https://doi.org/10.1109/InCIT56086.2022.10067382