



Research article

Modeling wind speed using Kumaraswamy–Weibull distribution via improved metaheuristic optimization algorithms: The Edremit Gulf, Turkey

Adil Kılıç¹, Gamze Güven², Özge Gürer^{3,*} and Birdal Şenoğlu³

¹ Department of Statistics, Kırıkkale University, Kırıkkale 71450, Turkey

² Department of Statistics, Eskisehir Osmangazi University, Eskisehir 26040, Turkey

³ Department of Statistics, Ankara University, Ankara 06100, Turkey

* **Correspondence:** Email: otanju@ankara.edu.tr.

Abstract: Wind speed is a key factor for assessing the wind energy potential of a region. Therefore, determining the best statistical distribution for modeling wind speed data is crucial for effective resource assessment and optimized energy production. The Weibull distribution is widely used for this purpose; however, it may not always provide the most adequate fit. In this study, Kumaraswamy–Weibull (KumW) distribution is used to model wind speed data as a more flexible alternative to Weibull distribution. In estimating the parameters of the KumW distribution, maximum likelihood (ML) methodology is employed. The resulting likelihood equations are not solvable explicitly because they involve nonlinear functions of unknown parameters. In order to obtain the ML estimates of the KumW parameters, this study utilizes three popular metaheuristic algorithms, the genetic algorithm (GA), particle swarm optimization (PSO), and differential evolution (DE), which have emerged as effective alternatives to the traditional numerical methods for solving computationally challenging optimization problems. One of the key challenges in using metaheuristic algorithms is the proper determination of an effective search space. In the literature, search spaces are often chosen arbitrarily, which may lead to reduced accuracy. As a novel contribution, we propose a new search space based on the bootstrap confidence intervals of the unknown KumW parameters for GA, PSO, and DE algorithms. The performances of the estimators obtained using GA, PSO, and DE based on proposed search space are compared with the corresponding estimators based on fixed search space in terms of bias, mean squares error (MSE), and deficiency (Def) criteria via an extensive Monte Carlo simulation study. Finally, the annual and seasonal wind speed data obtained from two stations located in the northern part of the Edremit Gulf, Turkey are modeled using the KumW distribution to demonstrate the practical utility of the proposed search space. The fitting performance of KumW is compared with that of the well-known and widely used Weibull distribution with respect to the root mean square error (RMSE), the mean absolute error (MAE), the coefficient of determination (R^2), the Akaike information (AIC) and Kolmogorov–Smirnov (KS) criteria. Moreover,

the power density error (PDE) criterion is utilized to evaluate the capability of KumW distribution in estimating wind power.

Keywords: wind speed modeling; Kumaraswamy–Weibull distribution; metaheuristic algorithms; search space; Monte Carlo simulation

1. Introduction

Global energy demand steadily increases due to population growth, industrialization, and technological advancements. Fossil fuels are the dominant source of global energy production; however, their limited reserves and detrimental environmental impacts necessitate a transition to more sustainable alternatives. Among these, wind energy stands out as a clean and abundant resource, increasingly adopted worldwide. The efficient utilization of wind energy requires optimal placement of wind turbines and wind farms, which in turn depends on a comprehensive understanding of wind characteristics. One of the most important factors in assessing a region's wind energy potential is wind speed. Therefore, accurate determination of the wind speed distribution is essential for effectively evaluating wind resources and optimizing energy production. Beyond wind energy assessment, accurate characterization of a region's wind-speed regime is also important for various wind-engineering applications, such as wind-resistant structural design and the probabilistic assessment of wind-induced loads and structural responses, for which wind speed constitutes one of the key inputs; see, for example, [1–4].

In this context, selecting an appropriate probability distribution for wind speed is important for both wind energy assessment and wind engineering applications. The Weibull distribution is widely used for modeling wind speed data; see, for example, [5–8]. However, it may be inadequate in providing an accurate fit in many cases. Hence, there exist numerous studies employing alternative distributions for better modeling wind speed data and therefore enhancing modeling precision. For example, Jaramillo and Borja [9] developed a model based on a bimodal Weibull-Weibull probability density function to model wind speed distribution in La Ventosa, Oaxaca, Mexico. Akpınar and Akpınar [10] developed a maximum-entropy-principle-type exponential family of distribution functions for describing wind speed density and showed that the proposed family performs better than the Weibull distribution. Brano et al. [11] compared seven probability density functions (PDFs) for fitting wind speed data in the urban area of Palermo, Italy, and they concluded that the Burr distribution is the most reliable when compared to others. Philippopoulos et al. [12] proposed using gamma distribution as an alternative to the Weibull distribution for modeling the experimental wind speed frequency distributions for the monthly and overall period in the greater area of a Mediterranean coastal valley in the island of Crete, Greece. Drobinski et al. [13] compared several probability distributions for modeling wind speed with the traditional Weibull model using data from 89 stations in France and demonstrated that the Rayleigh–Rice distribution provides the most accurate fit across diverse meteorological and topographic conditions. Akgül et al. [14] proposed the inverse Weibull (IW) distribution as an alternative to the Weibull distribution for modeling wind speed data, particularly those having extreme or outlying wind speeds in the direction of the right tail. Alavi et al. [15] utilized the Nakagami distribution for the first time to estimate wind speed distribution and

compared it with seven more common distributions. Chiodo and De Falco [16] suggested an inverse Burr distribution as a useful alternative for probabilistic modeling of extreme wind speed values. Arslan et al. [17] proposed the generalized Lindley (GL) and power Lindley (PL) distributions as alternatives to the commonly used Weibull distribution for modeling wind speed data, demonstrating that the GL distribution provides the best fit in terms of statistical criteria, whereas the PL distribution is preferable for minimizing the power density error. Aries et al. [18] compared eight distributions for modeling wind speed data at four locations in Algeria and stated that the generalized extreme value and gamma distributions are the most appropriate among others. Bagci et al. [19] used inverted Kumaraswamy distribution for the first time to model wind speed data as an alternative to the Weibull distribution. Ahsan ul Haq et al. [20] modeled wind speed data by using the Marshall–Olkin length-biased exponential distribution.

In this study, the Kumaraswamy–Weibull (KumW) distribution, a generalization of the Weibull distribution proposed by Cordeiro et al. [21], is employed to model wind speed data. The KumW is a highly flexible, four-parameter distribution and therefore provides a better fit than its important submodels, including the Weibull, exponentiated Weibull (EW), and exponentiated exponential (EE); see, for example, [21] in the context of reliability.

Maximum likelihood (ML) estimation is one of the most widely used methods for estimating distribution parameters because ML estimators possess some desirable asymptotic properties under regularity conditions. In this study, we therefore employ ML methodology to estimate the parameters of the KumW distribution. However, due to the complex four-parameter structure of the KumW distribution, its likelihood equations involve nonlinear functions of the distribution parameters. As a result, it is not possible to derive closed-form solutions for ML estimators. In such cases, traditional numerical methods such as the Newton–Raphson, Broyden–Fletcher–Goldfarb–Shanno (BFGS), and Nelder–Mead algorithms are commonly utilized. However, these methods have several limitations, including sensitivity to initial values, risk of converging to local optima, and poor performance in many complex problems. To overcome these problems, metaheuristic algorithms are increasingly considered as alternatives to conventional numerical methods. In this study, we employ three well-known metaheuristic algorithms: the genetic algorithm (GA), particle swarm optimization (PSO), and differential evolution (DE) due to their widespread usage; see, for example, [22–29]. Numerous studies also used these algorithms to model wind speed data. For example, Chang [30] applied PSO to estimate the Weibull distribution parameters for wind speed modeling in Taiwan, demonstrating its efficiency in improving parameter estimation accuracy and assessing wind energy potential across different climatic regions. Liu et al. [31] employed GA to estimate Weibull parameters for wind characterization analysis in the Taiwan Strait. Lydia et al. [32] used the DE algorithm to estimate the wind turbine power curve modeled with a five-parameter logistic function. Carneiro et al. [33] evaluated the effectiveness of PSO in estimating Weibull parameters for wind resource assessment in northeast Brazil and demonstrated that PSO outperforms conventional numerical methods with higher accuracy and lower estimation errors. Kromer et al. [34] applied the DE algorithm to fit a mixed Weibull distribution to wind speed data and evaluated the effectiveness of various DE variants. Kumar and Kumar [35] introduced an approach that utilizes the DE algorithm to enhance the accuracy of parameter estimation for the Weibull distribution in wind speed data modeling applications.

Proper determination of the search space plays a crucial role in improving the performance of GA,

PSO, and DE algorithms. In the literature, search spaces are often defined arbitrarily and tend to be too broad due to the lack of knowledge about the true parameter values. However, an excessively large search space can considerably reduce the efficiency of the algorithms. A notable contribution of this study is to propose a new search space based on bootstrap confidence intervals (CIs) to improve the performance of GA, PSO, and DE algorithms in estimating the unknown parameters of the KumW distribution with higher accuracy and lower estimation error. Also note that several adaptive and reduced search space strategies have been proposed in the literature. For example, Amirjanov [36] investigated a changing-range GA that shifts and shrinks the search space using feasible and infeasible solutions from the evolving population. Adewumi and Arasomwan [37] proposed an improved PSO that adjusts the search space and velocity limits at each iteration based on the extreme coordinate values of the particles. Luo et al. [38] proposed a sampling-based adaptive bounding evolutionary algorithm that updates the search space during evolution using information from individuals selected over multiple generations. These studies provide valuable strategies for adapting or reducing the search space. However, these approaches rely on information generated during optimization rather than directly using statistical information from the observed sample to determine the search space. More closely related to statistical parameter estimation, Yalçinkaya et al. [25] used confidence intervals based on modified ML (MML) estimators to define a reduced search space for GA in estimating the parameters of the skew-normal distribution. Ouédraogo et al. [39] proposed data-dependent bounds for the parameters of the three-parameter Weibull distribution and used the resulting search space for the DE algorithm in obtaining the ML estimates of the unknown distribution parameters. Although these approaches provide informative data-dependent search spaces, their construction requires distribution-specific analytical results, such as confidence intervals derived from MML estimators or theoretical bounds based on properties of order statistics. In contrast, the approach proposed in the present study does not require analytically derived standard errors or distribution-specific theoretical derivations of parameter bounds, including those based on properties of order statistics. It provides a bootstrap-based framework that is practical to implement for constructing data-driven search spaces. To demonstrate the effectiveness of this data-driven search space, the estimators obtained using GA, PSO, and DE based on bootstrap CIs are compared to their counterparts based on a fixed search space in terms of both bias and efficiency via a comprehensive Monte Carlo simulation study.

In the application part of this study, hourly mean wind speed data obtained from two stations located in the northern part of the Edremit Gulf, Turkey are modeled with the KumW distribution. The modeling performance of the KumW is then compared with the traditional Weibull distribution with respect to the root mean square error (RMSE), the mean absolute error (MAE), the coefficient of determination (R^2), the Akaike information (AIC), Kolmogorov–Smirnov (KS), and power density error (PDE) criteria.

The remaining parts of this study are organized as follows. In Section 2, the KumW distribution; the corresponding likelihood equations for its unknown parameters; and the metaheuristic algorithms GA, PSO, and DE are introduced. The steps of the proposed search space are given in Section 3. In Section 4, the results of the Monte Carlo simulation study are discussed. In Section 5, the details of the wind speed data analysis and the corresponding results are presented. Finally, concluding remarks are given in Section 6.

2. Materials and methods

This section provides the necessary theoretical background for the use of metaheuristic algorithms in estimating the parameters of the KumW distribution. First, a brief description of the KumW distribution is given in Subsection 2.1. Then, ML estimation procedure for the parameters of KumW distribution, that is, a , b , p , and σ , is provided in Subsection 2.2. The details of the GA, PSO, and DE algorithms are presented in Subsection 2.3.

2.1. KumW distribution

The KumW distribution extends the classical Weibull distribution by introducing additional shape parameters. As a result, it offers greater flexibility and better modeling performance for the data sets with varying levels of skewness and kurtosis.

The PDF and cumulative distribution function (CDF) of the KumW distribution are given as follows:

$$f_X(x) = \frac{abp}{\sigma} \left(\frac{x}{\sigma}\right)^{p-1} \exp\left\{-\left(\frac{x}{\sigma}\right)^p\right\} \left[1 - \exp\left\{-\left(\frac{x}{\sigma}\right)^p\right\}\right]^{a-1} \left\{1 - \left[1 - \exp\left\{-\left(\frac{x}{\sigma}\right)^p\right\}\right]^a\right\}^{b-1}, \quad (2.1)$$

$$x > 0; \sigma > 0; a, b, p > 0$$

and

$$F_X(x) = 1 - \left\{1 - \left[1 - \exp\left\{-\left(\frac{x}{\sigma}\right)^p\right\}\right]^a\right\}^b, \quad (2.2)$$

respectively. Here, a , b , and p are the shape parameters, and σ is the scale parameter. A random variable X following KumW distribution is denoted by $X \sim KumW(a, b, p, \sigma)$. Figure 1 presents the PDF plots of the KumW distribution for some representative values of the shape parameters to provide better insight into its behavior.

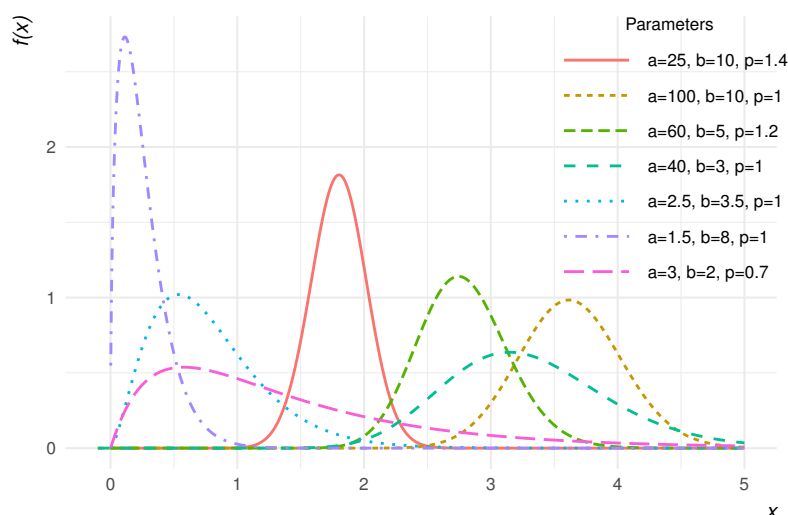


Figure 1. The PDF plots of the $KumW(a, b, p, \sigma = 1)$ distribution.

See also Table 1 for its skewness ($\sqrt{\beta_1}$) and kurtosis (β_2) values.

Table 1. Skewness and kurtosis values of the $KumW(a, b, p, \sigma = 1)$ distribution.

(a, b, p)	$(25, 10, 1.4)$	$(100, 10, 1)$	$(60, 5, 1.2)$	$(40, 3, 1)$	$(2.5, 3.5, 1)$	$(1.5, 8, 1)$	$(3, 2, 0.7)$
$\sqrt{\beta_1}$	0.01	0.10	0.25	0.55	1.08	1.35	2.01
β_2	3.02	3.05	3.18	3.63	4.74	5.62	9.94

Realize that KumW contains many important distributions as its submodels. For example, it reduces to the Weibull for $a = b = 1$, exponentiated Weibull for $b = 1$, exponential for $a = b = p = 1$, exponentiated exponential for $b = p = 1$, Rayleigh for $a = b = 1, p = 2$, exponentiated Rayleigh for $b = 1, p = 2$, and so forth. See Cordeiro et al. [21] for the details of the KumW distribution.

2.2. Maximum likelihood estimation

Let $X_1, X_2, \dots, X_n$ be a random sample from $KumW(a, b, p, \sigma)$. Then, the log-likelihood ($\ln L$) function is obtained as

$$\begin{aligned} \ln L = & n \ln(abp) - n \ln \sigma + (p-1) \sum_{i=1}^n \ln\left(\frac{x_i}{\sigma}\right) - \sum_{i=1}^n \left(\frac{x_i}{\sigma}\right)^p \\ & + (a-1) \sum_{i=1}^n \ln u_i + (b-1) \sum_{i=1}^n \ln\{1 - u_i^a\}, \end{aligned} \quad (2.3)$$

where $u_i = [1 - \exp\{-(x_i/\sigma)^p\}]$. The score functions are then obtained by differentiating $\ln L$ with respect to the parameters a, b, p , and σ as follows:

$$\begin{aligned} \frac{\partial \ln L}{\partial a} &= \frac{n}{a} + \sum_{i=1}^n \ln u_i - (b-1) \sum_{i=1}^n \frac{u_i^a \ln u_i}{1 - u_i^a}, \\ \frac{\partial \ln L}{\partial b} &= \frac{n}{b} + \sum_{i=1}^n \ln(1 - u_i^a), \\ \frac{\partial \ln L}{\partial p} &= \frac{n}{p} + \sum_{i=1}^n \ln\left(\frac{x_i}{\sigma}\right) - \sum_{i=1}^n \left(\frac{x_i}{\sigma}\right)^p \ln\left(\frac{x_i}{\sigma}\right) + (a-1) \sum_{i=1}^n \frac{(1 - u_i) \left(\frac{x_i}{\sigma}\right)^p \ln\left(\frac{x_i}{\sigma}\right)}{u_i} \\ &\quad - (b-1) \sum_{i=1}^n \frac{a u_i^{a-1} (1 - u_i) \left(\frac{x_i}{\sigma}\right)^p \ln\left(\frac{x_i}{\sigma}\right)}{1 - u_i^a}, \end{aligned}$$

and

$$\begin{aligned} \frac{\partial \ln L}{\partial \sigma} &= -\frac{np}{\sigma} + \frac{p}{\sigma} \sum_{i=1}^n \left(\frac{x_i}{\sigma}\right)^p - \frac{p(a-1)}{\sigma} \sum_{i=1}^n \frac{\left(\frac{x_i}{\sigma}\right)^p (1 - u_i)}{u_i} \\ &\quad + \frac{pa(b-1)}{\sigma} \sum_{i=1}^n \frac{u_i^{a-1} \left(\frac{x_i}{\sigma}\right)^p (1 - u_i)}{1 - u_i^a}, \end{aligned}$$

respectively. ML estimators for a , b , p , and σ are obtained by solving the system of likelihood equations $\partial \ln L / \partial a = 0$, $\partial \ln L / \partial b = 0$, $\partial \ln L / \partial p = 0$, and $\partial \ln L / \partial \sigma = 0$. Because these equations involve nonlinear terms, they do not admit closed-form solutions. Iterative methods are generally used when the ML estimators of distribution parameters cannot be obtained in explicit forms, as in the case for KumW distribution. In this study, we employ widely used population based metaheuristic algorithms GA, PSO, and DE because they are well-established and extensively used in complex optimization problems and serve as effective alternatives to traditional numerical methods.

2.3. Metaheuristic algorithms

Metaheuristic algorithms are inspired by natural processes such as evolution or swarm intelligence and mimic their behavior to efficiently search for near-optimal solutions. Among the broad class of metaheuristics, GA, PSO, and DE are widely applied in solving various complex optimization tasks. These methods iteratively evolve a set of candidate solutions by exploring the search space and exploiting the information gathered from previous iterations. The quality of each solution is evaluated using a predefined objective or fitness function, which guides the search toward more promising regions of the solution space. The operational steps of these three algorithms are described in the following subsections.

2.3.1. Genetic algorithm

GA, proposed by Holland [40], is inspired by the principles of natural selection and genetic evolution. The algorithm begins with an initial population (generation) of randomly generated candidate solutions (chromosomes). Each solution is evaluated using a fitness function that quantifies its performance with respect to the considered optimization problem. Genetic operators such as selection, crossover, and mutation are applied to guide the population toward better solutions. Selection favors individuals with higher fitness to contribute to the next generation, crossover merges information from two parent solutions to produce new candidates, and mutation introduces random changes to maintain diversity and prevent premature convergence. This process is repeated iteratively until a stopping criterion such as a maximum number of generations or convergence to an acceptable solution is satisfied.

In order to obtain the ML estimates for the parameters of the KumW distribution, that is, $\theta = (a, b, p, \sigma)$, by using GA, the following steps are performed:

Step 1. Identify the search space, and set the GA parameters: crossover probability (CP), mutation probability (MP), selection rate (SR), number of elite chromosomes (E), and population size (N).

Step 2. Generate the initial population of N chromosomes from the search space. In this study, the chromosomes in the population at iteration k are denoted as $(\theta_1^{(k)}, \theta_2^{(k)}, \dots, \theta_N^{(k)})$, where $\theta_i^{(k)} = (a_i^{(k)}, b_i^{(k)}, p_i^{(k)}, \sigma_i^{(k)})$, $i = 1, \dots, N$. Note that k is equal to 0 for the initial population.

Step 3. Evaluate the fitness value of each chromosome $\theta_i^{(k)}$ in the population at iteration k , that is, $\ln L(\theta_i^{(k)})$.

Step 4. Apply the elitism operator. E chromosomes with the highest fitness values are identified and directly transferred to the next generation without any modification.

Step 5. Select parent chromosomes from the remaining population (excluding elites) using a selection rule such as the roulette wheel, where the proportion of selected individuals is determined by SR.

Step 6. Apply the crossover operator to the selected parents according to CP to generate new offspring (individuals) by combining the parts of parents.

Step 7. Apply the mutation operator according to MP by randomly changing parts of offspring to maintain diversity in the population.

Step 8. Set $k = k + 1$, and repeat Steps 3–7 until a termination criterion is met (e.g., reaching a maximum number of iterations or convergence of the best fitness value).

2.3.2. Particle swarm optimization

PSO was introduced by Kennedy and Eberhart [41] and inspired by the social behavior of animals such as birds and fish. In PSO, each candidate solution is referred to a particle, and the group of all particles is known as a swarm. Particles move through the search space by adjusting their velocities and positions according to both their own best individual positions (personal best) and the best position known in the entire swarm (global best). This mechanism enables the swarm to collectively search for better solutions.

In order to obtain the ML estimates for the parameters of the KumW distribution, that is, $\theta = (a, b, p, \sigma)$, by using PSO, the following steps are performed:

Step 1. Identify the search space, and set the PSO parameters: inertia weight (w), acceleration coefficients (c_1, c_2), and swarm size (S).

Step 2. Generate the initial swarm of S particles from the search space. Positions of each particle i , $i = 1, \dots, S$, and the corresponding velocities are stored in vectors $\theta_i^{(k)} = (a_i^{(k)}, b_i^{(k)}, p_i^{(k)}, \sigma_i^{(k)})$ and $\mathbf{v}_i^{(k)} = (v_{a_i}^{(k)}, v_{b_i}^{(k)}, v_{p_i}^{(k)}, v_{\sigma_i}^{(k)})$, respectively. Here, k is the iteration number and is equal the 0 for the initial swarm.

Step 3. Evaluate the fitness value of each particle in the swarm at iteration k , that is, $\ln L(\theta_i^{(k)})$, $i = 1, 2, \dots, S$.

Step 4. Determine $\mathbf{pbest}_i^{(k)}$ and $\mathbf{gbest}^{(k)}$ according to the fitness function, where $\mathbf{pbest}_i^{(k)}$ denotes the vector storing the personal best position of particle i until the iteration k , and $\mathbf{gbest}^{(k)}$ represents the global best position among all particles at the iteration k .

Step 5. Update the velocity and position of each particle according to

$$\mathbf{v}_i^{(k+1)} = w\mathbf{v}_i^{(k)} + c_1r_1(\mathbf{pbest}_i^{(k)} - \theta_i^{(k)}) + c_2r_2(\mathbf{gbest}^{(k)} - \theta_i^{(k)}),$$

$$\theta_i^{(k+1)} = \theta_i^{(k)} + \mathbf{v}_i^{(k+1)},$$

where r_1 and r_2 are independent random numbers uniformly distributed in $[0, 1]$.

Step 6. Set $k = k + 1$, and repeat Steps 3–5 until a termination criterion is satisfied (e.g., reaching a maximum number of iterations or convergence of the global best fitness value).

2.3.3. Differential evolution

DE, proposed by Storn and Price [42], is designed for solving problems with real-valued parameters. It is a widely used algorithm that is appreciated for its simplicity and fast convergence. Similarly to GA, DE uses genetic operators such as mutation, crossover, and selection. However, the primary mechanism for generating new solutions differs between the two methods. GA generates new solutions mainly through crossover, whereas DE uses mutation as its core operator to create new solutions by combining the differences between randomly selected individuals from the population. The resulting solutions are then mixed with existing ones through crossover. After that, selection is applied between the current and newly generated solution, and the one with the better fitness value is carried forward to the next generation. By repeating this process over generations, DE steadily improves the population, leading to efficient convergence towards optimal or near-optimal solutions, even for complex problems.

In order to obtain the ML estimates for the parameters of the KumW distribution, that is, $\theta = (a, b, p, \sigma)$, by using DE, the following steps are performed:

Step 1. Identify the search space, and set the initial DE parameters: scale factor (F), crossover rate (CR), and population size (N).

Step 2. Generate the initial population of N individuals. The target vector corresponding to each individual is obtained as

$$\theta_{i,j}^{(k)} = \theta_{\text{low}} + r_{i,j}(\theta_{\text{up}} - \theta_{\text{low}}), \quad r_{i,j} \sim \mathcal{U}(0, 1), \quad i = 1, \dots, N, \quad j = 1, \dots, D,$$

where $D = 4$, and it corresponds to the number of unknown parameters (a, b, p, σ) in this study. $\theta_{\text{low}} = (a_{\min}, b_{\min}, p_{\min}, \sigma_{\min})$ and $\theta_{\text{up}} = (a_{\max}, b_{\max}, p_{\max}, \sigma_{\max})$ denote the componentwise lower and upper bounds, k is the iteration number, and is equal the 0 for the initial population.

Step 3. Evaluate the fitness value of the target vector corresponding to each individual in the population at iteration k , that is, $\ln L(\theta_i^{(k)})$, $i = 1, \dots, N$, where $\theta_i^{(k)} = (\theta_{i,1}^{(k)}, \theta_{i,2}^{(k)}, \theta_{i,3}^{(k)}, \theta_{i,4}^{(k)}) = (a_i^{(k)}, b_i^{(k)}, p_i^{(k)}, \sigma_i^{(k)})$.

Step 4. Apply the mutation operator to each target vector to construct the donor vector

$$\mathbf{v}_i^{(k)} = \theta_i^{(k)} + F(\theta_{\text{best}}^{(k)} - \theta_i^{(k)}) + F(\theta_{r_1}^{(k)} - \theta_{r_2}^{(k)}),$$

where $\theta_{\text{best}}^{(k)}$ is the current best individual, and r_1, r_2 are two distinct indices such that $r_1 \neq r_2 \neq 1$.

Step 5. Apply the crossover operator to combine the donor and target vectors, producing the trial vector $\mathbf{u}_{i,j}^{(k)}$ as

$$\mathbf{u}_{i,j}^{(k)} = \begin{cases} \mathbf{v}_{i,j}^{(k)}, & \text{if } \text{rand}_{i,j} \leq \text{CR or } j = j_{\text{rand}}, \\ \theta_{i,j}^{(k)}, & \text{otherwise,} \end{cases}$$

where $\text{rand}_{i,j} \sim \mathcal{U}(0, 1)$, and $j_{\text{rand}} \in \{1, \dots, 4\}$.

Step 6. Apply the selection operator. If $\ln L(\mathbf{u}_i^{(k)}) \geq \ln L(\theta_i^{(k)})$, set $\theta_i^{(k+1)} = \mathbf{u}_i^{(k)}$; otherwise, set $\mathbf{u}_i^{(k+1)} = \theta_i^{(k)}$, where $\mathbf{u}_i^{(k)} = (u_{i,1}^{(k)}, u_{i,2}^{(k)}, u_{i,3}^{(k)}, u_{i,4}^{(k)})$. Update the best solution $\theta_{\text{best}}^{(k+1)}$ accordingly.

Step 7. Set $k = k + 1$, and repeat Steps 3–6 until a termination criterion is satisfied (e.g., maximum number of iterations or convergence of the best fitness value).

3. Proposed search space

Metaheuristic algorithms are sensitive to the proper choice of their parameters such as population size, mutation and crossover rates, or inertia weight. Moreover, an appropriate determination of the search space plays a crucial role in their performance because it affects the efficiency and accuracy of these algorithms.

In many studies, the search space is determined arbitrarily, often based on prior information or subjective judgment. Researchers may define a search space that is considerably broad in order to increase the chances of finding a global optimum rather than getting stuck in local optima and to ensure that the true parameter values lie within the specified bounds. Although this approach is intended to prevent the exclusion of the optimal region, it often results in reduced accuracy.

To overcome this drawback, we propose a new search space based on the bootstrap CIs for the parameters of the KumW distribution. In doing so, we use a systematic and objective approach in which the search space bounds for each parameter are defined by its bootstrap confidence interval, while noting that these intervals may not necessarily include the ML estimates. As a result, the optimization process focuses on a data-driven search space. Although illustrated for the KumW distribution, the idea can be applied to any distribution.

Let the data set $\{x_1, x_2, \dots, x_n\}$ follow $\text{KumW}(a, b, p, \sigma)$, and let $\hat{a}, \hat{b}, \hat{p}, \hat{\sigma}$ denote the estimators of interest. To build the bootstrap-based search space, we use the following algorithm. Because the steps are the same for each parameter, we describe them for the parameter a only.

Algorithm Bootstrap-based search space construction

- Step 1. Generate a bootstrap sample $\{x_1^*, x_2^*, \dots, x_n^*\}$ by sampling with replacement from the data $\{x_1, \dots, x_n\}$.
- Step 2. Compute the bootstrap replication $\hat{a}^*$ of the estimator $\hat{a}$ for the bootstrap sample given in Step 1 using the least squares (LS) methodology.
- Step 3. Repeat Steps 1 and 2 B times to obtain bootstrap replications $\hat{a}_1^*, \hat{a}_2^*, \dots, \hat{a}_B^*$.
- Step 4. Sort the replications in ascending order: $\hat{a}_{(1)}^* < \hat{a}_{(2)}^* < \dots < \hat{a}_{(B)}^*$.
- Step 5. Form the $100(1 - \alpha)\%$ bootstrap percentile CI for the parameter a as

$$(\hat{a}_{(\alpha/2)}^*, \hat{a}_{(1-\alpha/2)}^*),$$

which satisfies

$$P(\hat{a}_{(\alpha/2)}^* < a < \hat{a}_{(1-\alpha/2)}^*) \approx 1 - \alpha,$$

where $\hat{a}_{(q)}^*$ is the empirical q th percentile of the bootstrap distribution.

- Step 6. Identify the search space for all parameters by taking their $100(1 - \alpha)\%$ bootstrap percentile CIs as bounds:

$$\begin{aligned} a &\in (\hat{a}_{(\alpha/2)}^*, \hat{a}_{(1-\alpha/2)}^*), \\ b &\in (\hat{b}_{(\alpha/2)}^*, \hat{b}_{(1-\alpha/2)}^*), \\ p &\in (\hat{p}_{(\alpha/2)}^*, \hat{p}_{(1-\alpha/2)}^*), \\ \sigma &\in (\hat{\sigma}_{(\alpha/2)}^*, \hat{\sigma}_{(1-\alpha/2)}^*). \end{aligned}$$

Note that the LS methodology employed in Step 2 proceeds as follows. For the bootstrap sample $\{x_1^*, \dots, x_n^*\}$ given in Step 1, the observations are first ordered as $x_{(1)}^* \leq x_{(2)}^* \leq \dots \leq x_{(n)}^*$. The bootstrap replications $\hat{a}^*$, $\hat{b}^*$, $\hat{p}^*$, and $\hat{\sigma}^*$ are then obtained by minimizing the LS objective function

$$\sum_{i=1}^n \left[F_X(x_{(i)}^*) - \frac{i}{n+1} \right]^2 \quad (3.1)$$

with respect to a, b, p , and σ , respectively. Here, $F_X(\cdot)$ is the CDF of the KumW distribution given in (2.2), and $i/(n+1)$, ($i = 1, \dots, n$), denote the expected values of $F_X(x_{(i)}^*)$.

4. Monte Carlo simulation study

In this section, a comprehensive Monte Carlo simulation study is conducted to evaluate the effectiveness of the proposed search space on the metaheuristic algorithms considered in this study. The performances of GA, PSO, and DE algorithms based on the proposed search space (PSS) are compared with their conventional counterparts based on a fixed search space (FSS) for estimating the unknown parameters of the KumW distribution. For brevity, the PSS-based algorithms are hereafter denoted as GA(PSS), PSO(PSS), and DE(PSS) and their FSS-based counterparts as GA(FSS), PSO(FSS), and DE(FSS).

PSS for KumW parameters a, b, p , and σ are identified by following the steps in the algorithm given in Section 3. On the other hand, FSS is arbitrarily determined within broad but plausible bounds, which are assumed to contain the true parameter values based on prior evidence from the literature. The PSS and FSS for the shape parameters a, b , and p and the scale parameter σ of the KumW distribution are identified as follows:

PSS:	Bootstrap percentile CIs described in Step 6 of the algorithm in Section 3.
FSS:	$[1, 10]$ for a , $[1, 10]$ for b , $[0.1, 10]$ for p and $[0.1, 10]$ for σ .

In constructing the PSS, bootstrap resampling with $B = 1000$ replications for each sample size n is used. The significance level is set to $\alpha = 0.05$, corresponding to 95% bootstrap percentile intervals.

We consider the following parameter settings:

Case I: $(a, b, p) = (1.5, 8, 1.5)$

Case II: $(a, b, p) = (1.8, 6, 0.9)$

Case III: $(a, b, p) = (2, 4, 0.8)$

Case IV: $(a, b, p) = (3, 2, 0.7)$

for the KumW distribution to comprehensively evaluate the performance of the proposed search space. The scale parameter σ is taken to be 1 without loss of generality. Cases I–IV are designed to represent positively skewed and leptokurtic distributions, as commonly observed in wind speed data; see, for example, [43–46]. It should also be noted that the cases are determined such that both skewness and kurtosis values of the KumW distribution increase progressively from Case I to Case IV.

By combining each case (I–IV) with different sample sizes ($n = 100, 500, 1000, 2500, 5000$), a total of $4 \times 5 = 20$ scenarios are considered in the simulation study. Table 2 gives the simulation setup for the parameters of metaheuristic algorithms considered in this study.

Table 2. Simulation setup for the parameters of the metaheuristic algorithms.

GA:	$CP = 0.8, MP = 0.2, E = 3, N = 50, maxit = 100$
PSO:	$w = 0.72, c_1 = c_2 = 1.19, S = 50, maxit = 100$
DE:	$F = 0.8, CR = 0.5, N = 50, maxit = 100$

Here, *maxit* denotes the maximum number of iterations. The resulting estimators for the unknown parameters of the KumW distribution are denoted by $\hat{a}$, $\hat{b}$, $\hat{p}$, and $\hat{\sigma}$. To compare the performances of the GA, PSO, and DE based on the PSS and FSS in estimating KumW parameters, we employ the bias, mean squares error (MSE), and deficiency (Def) criteria defined as

$$Bias = \frac{1}{s} \sum_{i=1}^s \hat{\theta}_i - \theta, \quad (4.1)$$

$$MSE = \frac{1}{s} \sum_{i=1}^s (\hat{\theta}_i - \theta)^2, \quad (4.2)$$

and

$$Def = MSE(\hat{a}) + MSE(\hat{b}) + MSE(\hat{p}) + MSE(\hat{\sigma}), \quad (4.3)$$

where s represents the number of Monte Carlo replications and is fixed at 1000 across all scenarios. Here, θ is the parameter of interest, and $\hat{\theta}_i$ is the estimated value of the parameter θ obtained from the i th replication.

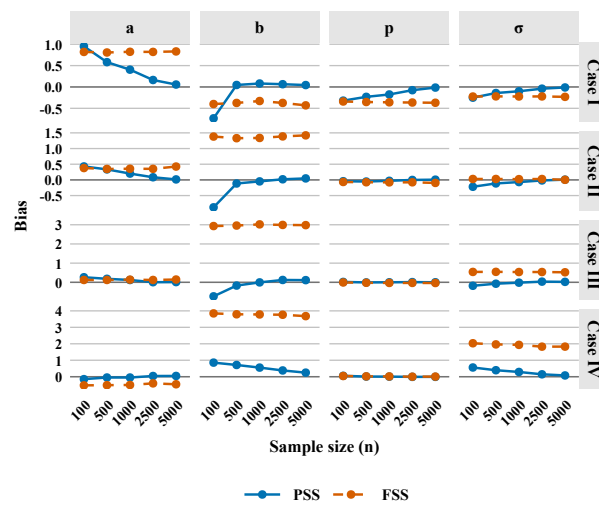
To provide fair and consistent comparisons for the bias patterns of GA, PSO, and DE based on the PSS and FSS, line plots are used; see Figure 2.

Moreover, simulation results regarding the MSE and Def values are shown as heatmaps to provide a visual representation of the performance differences between the search spaces and metaheuristic algorithms. The heatmaps corresponding to MSE values are given in Figures 3–5, whereas the Def values are summarized in Figure 6.

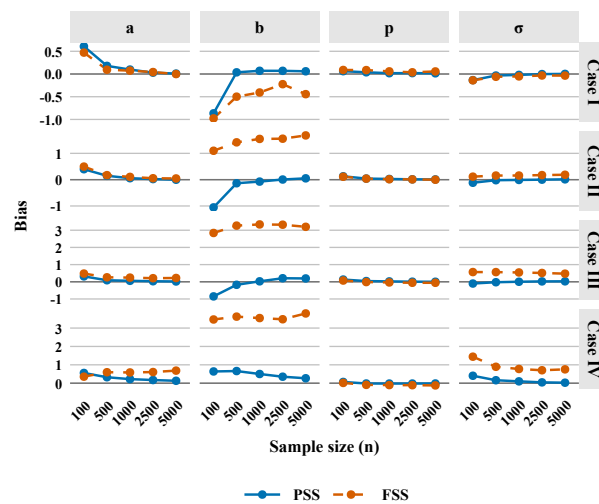
It should be noted that lower MSE and Def values indicate better performance; therefore, lighter colors in the heatmaps represent higher efficiency. The interpretation of the simulation results is organized in the following subsections.

4.1. Comparing the performances of search spaces PSS and FSS for each of the metaheuristic algorithms GA, PSO, and DE in terms of bias

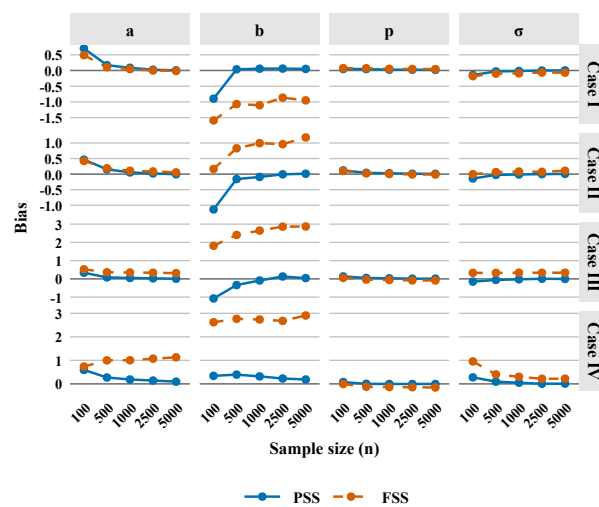
The results drawn from Figure 2 regarding the simulation study on bias comparisons are summarized in tabular form below for the sake of clarity and interpretability; see Table 3.



(a) Bias patterns for GA under the PSS and FSS.



(b) Bias patterns for PSO under the PSS and FSS.



(c) Bias patterns for DE under the PSS and FSS.

Figure 2. Bias patterns for algorithms under the PSS and FSS.

Table 3. Comparison of bias patterns for GA, PSO, and DE based on the PSS and FSS in estimating KumW parameters.

	GA	PSO	DE
Case I	<i>a</i> The difference between the biases of GA(PSS) and GA(FSS) is negligible when $n = 100$, whereas GA(PSS) exhibits less bias for $n > 100$.	PSO(PSS) and PSO(FSS) exhibit comparable bias performances for $n \geq 1000$, whereas PSO(FSS) performs slightly better for $n < 1000$.	DE(FSS) has lower bias at $n = 100$, whereas DE(PSS) and DE(FSS) exhibit comparable bias performance for $n > 100$.
	<i>b</i> GA(FSS) has lower absolute bias at $n = 100$, whereas GA(PSS) performs better for $n > 100$.	PSO(PSS) and PSO(FSS) exhibit comparable bias performance at $n = 100$, whereas PSO(PSS) has lower absolute bias for $n > 100$.	DE(PSS) outperforms DE(FSS).
	<i>p</i> GA(PSS) shows better performance than GA(FSS) except when $n = 100$, where their performances are almost the same.	PSO(PSS) and PSO(FSS) show similar performance.	DE(PSS) and DE(FSS) show similar performance.
	σ GA(PSS) and GA(FSS) exhibit comparable absolute bias at $n = 100$, whereas GA(PSS) has lower absolute bias for $n \geq 500$.	Performances of PSO(PSS) and PSO(FSS) are nearly identical.	Performances of DE(PSS) and DE(FSS) are nearly identical.
Case II	<i>a</i> GA(PSS) and GA(FSS) exhibit comparable bias at $n = 100$ and 500, whereas GA(PSS) has lower bias for $n \geq 1000$.	PSO(PSS) and PSO(FSS) exhibit comparable performance.	DE(PSS) and DE(FSS) exhibit comparable performance.
	<i>b</i> GA(PSS) consistently outperforms GA(FSS) in terms of absolute bias.	PSO(PSS) outperforms PSO(FSS).	DE(FSS) has lower absolute bias than DE(PSS) when $n = 100$, whereas DE(PSS) outperforms DE(FSS) when the sample size is greater than 100.
	<i>p</i> GA(PSS) and GA(FSS) exhibit comparable performance.	PSO(PSS) and PSO(FSS) exhibit comparable performance.	DE(PSS) and DE(FSS) exhibit comparable performance.
	σ GA(FSS) has lower absolute bias than GA(PSS) for $n \leq 1000$, whereas their biases are comparable for $n \geq 2500$.	PSO(PSS) shows a slight performance advantage over PSO(FSS) except for $n = 100$, where they demonstrate similar performances.	DE(PSS) shows a slight performance advantage over DE(FSS) except when $n = 100$.
Case III	<i>a</i> GA(FSS) has lower bias than GA(PSS) at $n = 100$; their absolute biases are comparable at $n = 500$ and 1000, and GA(PSS) has lower bias for $n \geq 2500$.	PSO(PSS) performs slightly better than PSO(FSS).	DE(PSS) performs slightly better than DE(FSS).
	<i>b</i> GA(PSS) exhibits substantially higher performance than GA(FSS).	PSO(PSS) exhibits substantially higher performance than PSO(FSS).	DE(PSS) exhibits substantially higher performance than DE(FSS).
	<i>p</i> GA(PSS) exhibits almost the same performance with GA(FSS).	PSO(PSS) exhibits almost the same performance with PSO(FSS).	DE(PSS) exhibits almost the same performance with DE(FSS).
	σ GA(PSS) performs better than GA(FSS).	PSO(PSS) performs better than PSO(FSS).	DE(PSS) performs better than DE(FSS).
Case IV	<i>a</i> GA(PSS) performs better than GA(FSS).	PSO(PSS) performs better than PSO(FSS), except for $n = 100$, where PSO(FSS) performs slightly better.	DE(PSS) significantly outperforms DE(FSS), especially when the sample size is greater than 100.
	<i>b</i> GA(PSS) significantly outperforms GA(FSS).	PSO(PSS) significantly outperforms PSO(FSS).	DE(PSS) significantly outperforms DE(FSS).
	<i>p</i> Performances of GA(PSS) and GA(FSS) are nearly identical.	PSO(PSS) and PSO(FSS) exhibit comparable performance.	DE(FSS) performs slightly better than DE(PSS) at $n = 100$, whereas DE(PSS) performs slightly better for $n \geq 500$.
	σ The performance of GA(PSS) is substantially superior to that of GA(FSS).	PSO(PSS) performs better than PSO(FSS).	DE(PSS) performs better than DE(FSS) for all considered sample sizes, with the advantage being most pronounced at $n = 100$.

4.2. Comparing the performances of search spaces PSS and FSS for each of the metaheuristic algorithms GA, PSO, and DE in terms of MSE

In this subsection, the performance of each algorithm is assessed by comparing its PSS-based and FSS-based versions with respect to MSE values.

• Results for GA

For parameter a : When $n = 100$, GA(FSS) yields lower MSE in all cases. When $n = 500$, GA(PSS) performs better in Cases I and IV, whereas GA(FSS) performs better in Cases II and III. For $n \geq 1000$, GA(PSS) exhibits superior performance for all cases.

For parameter b : GA(PSS) significantly outperforms GA(FSS) in terms of MSE for all scenarios, with the performance advantage being particularly pronounced in Cases III and IV.

For parameter p : GA(FSS) yields lower MSE at $n = 100$ in all cases, whereas GA(PSS) yields lower MSE for $n \geq 500$.

For parameter σ : Except for Cases I and II when $n = 100$, GA(PSS) performs better than GA(FSS) in terms of MSE in all remaining settings.

• Results for PSO

For parameter a : Except for Cases I and IV when $n = 100$, and Case I when $n = 500$, PSO(PSS) has better performance than PSO(FSS) for all scenarios.

For parameters b and σ : PSO(PSS) substantially outperforms PSO(FSS) for all cases and sample sizes.

For parameter p : Consistent with the results obtained for parameter a , PSO(PSS) is preferable except for Cases I and IV when $n = 100$ and Case I when $n = 500$.

• Results for DE

For parameter a : Except for Case I when $n \leq 2500$; Cases II, III, and IV when $n = 100$; and Case II when $n = 500$, DE(PSS) performs better than DE(FSS) for all scenarios.

For parameters b : DE(PSS) substantially outperforms DE(FSS) in terms of MSE for all scenarios, with the performance advantage being particularly pronounced in Cases III and IV.

For parameter p : For Case I, DE(FSS) performs better than DE(PSS) except when $n = 5000$. For Cases II, III, and IV, DE(PSS) outperforms its rival except for Cases II and IV with $n = 100$ and for Case II with $n = 500$.

For parameter σ : DE(PSS) consistently yields lower MSE than DE(FSS) for all scenarios.

General comments

The simulation results reveal that, for parameters b and σ , efficiencies of the FSS-based GA, PSO, and DE algorithms decline dramatically for all sample sizes in Cases III and IV, where skewness and kurtosis values are high. It should be noted that PSS-based GA, PSO, and DE algorithms show a clear superiority over their FSS-based counterparts, especially in estimating the parameter b . As a general conclusion, although PSS-based GA, PSO, and DE algorithms demonstrate satisfactory performance even with smaller sample sizes, they are especially recommended for larger samples.

Because GA, PSO, and DE algorithms based on the PSS are generally found to be more efficient than their FSS-based counterparts, metaheuristic algorithms based on the PSS are compared with each other in the following subsection.

4.3. Comparing the performances of GA, PSO, and DE algorithms based on the PSS in terms of MSE

In this subsection, the performance of the PSS-based algorithms is evaluated using MSE values.

For parameter a : In Case I, DE and PSO perform comparably and yield the lowest MSE values except when $n = 100$, where GA performs best. In Cases II and III, GA performs the best for $n = 100$ and $n = 500$. For larger sample sizes, PSO and DE again show comparable performance and generally provide the lowest MSE values in both cases. In Case III, GA also becomes comparable to them when $n = 2500$ and $n = 5000$. In Case IV, GA consistently exhibits the best performance.

For parameter b : GA consistently yields the highest efficiency across all cases and sample sizes.

For parameter p : GA generally provides the lowest MSE values in Case I. For $n = 1000$, DE performs very similarly to GA, whereas for $n = 5000$, DE and PSO are comparable and yield the best results. In Case II, GA performs best for $n \leq 1000$, whereas PSO and DE become comparable and provide the lowest MSE values for larger sample sizes. In Cases III and IV, GA consistently gives the best performance.

For parameter σ : In Case I, GA yields the lowest MSE when $n = 100$, whereas PSO and DE show comparable and the best performance for larger sample sizes. In Case II, GA performs the best for $n = 100$ and $n = 500$. For $n = 1000$, the three algorithms show comparable performance, whereas PSO and DE yield the lowest MSE values for $n = 2500$ and $n = 5000$. In Case III, GA provides the highest efficiency except when $n = 2500$, where PSO performs best. In Case IV, GA performs the best for $n = 100$, whereas DE yields the lowest MSE values for the remaining sample sizes.

General comments

Overall, GA generally shows the best performance across the considered scenarios. The superiority of GA is especially evident for the parameter b , and it also generally performs the best for p . For a and σ , however, PSO and DE become more competitive and often outperform GA with comparable MSE values, particularly for larger sample sizes. Thus, although no single algorithm performs best in every setting, it can be concluded from these results that GA provides the best overall performance under PSS.

4.4. Comparing the performances of the search spaces and PSS-based metaheuristic algorithms in terms of Def

To evaluate the joint efficiencies of the estimators $\hat{a}$, $\hat{b}$, $\hat{p}$, and $\hat{\sigma}$ obtained by using the GA, PSO, and DE algorithms based on the PSS and FSS, the Def criterion is used as mentioned before.

The following results are derived from the heatmap shown in Figure 6, which is based on the Def values.

PSS-based algorithms substantially outperform their FSS-based counterparts in all considered scenarios except for GA in Case I when $n = 100$. PSS-based algorithms are also compared among themselves, as in the case of MSE. The results show that GA consistently achieves the best performance, except for Case I with $n > 100$, where PSO and DE perform similarly and outperform GA.

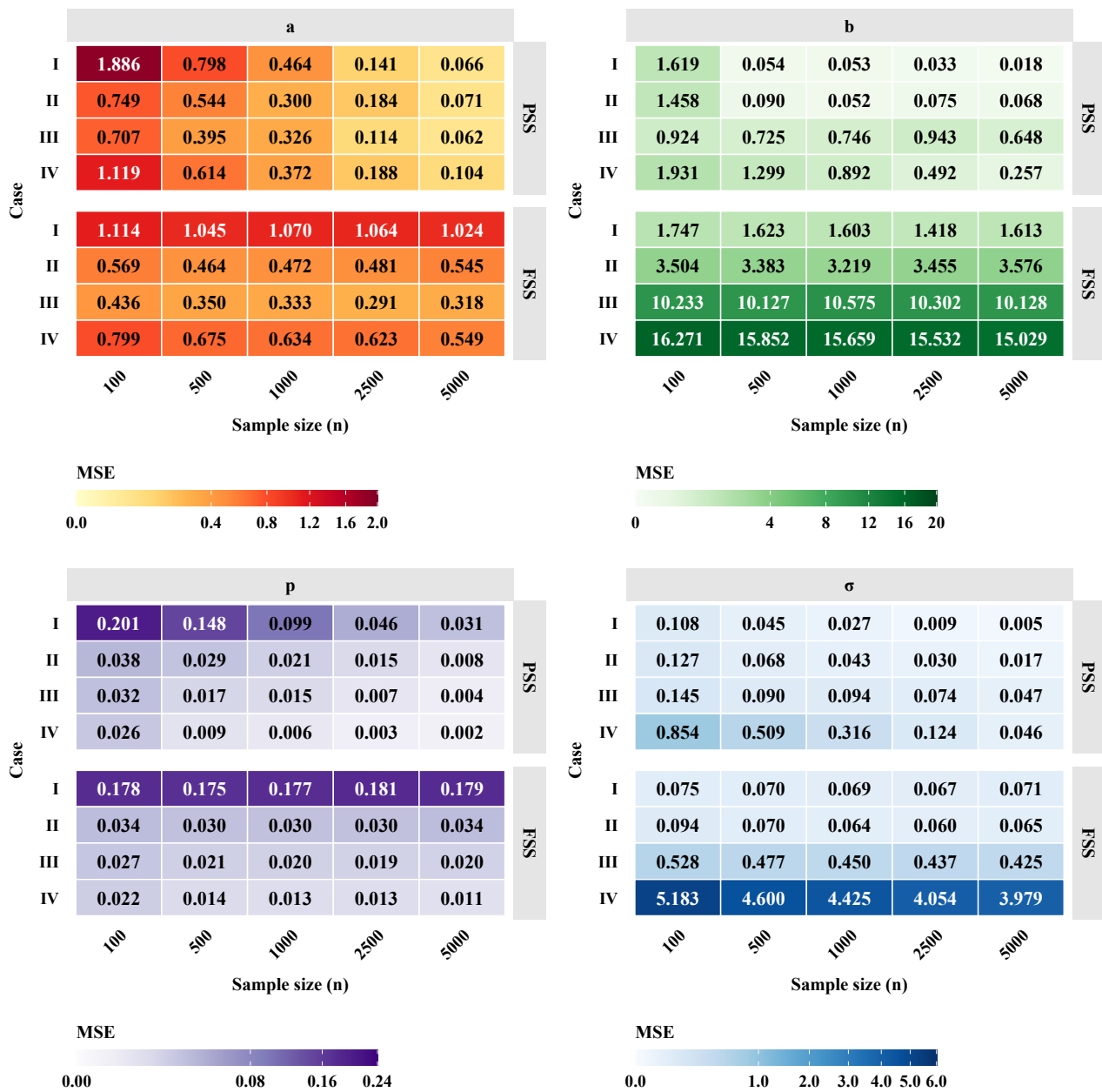


Figure 3. MSE heatmaps for GA under the PSS and FSS.

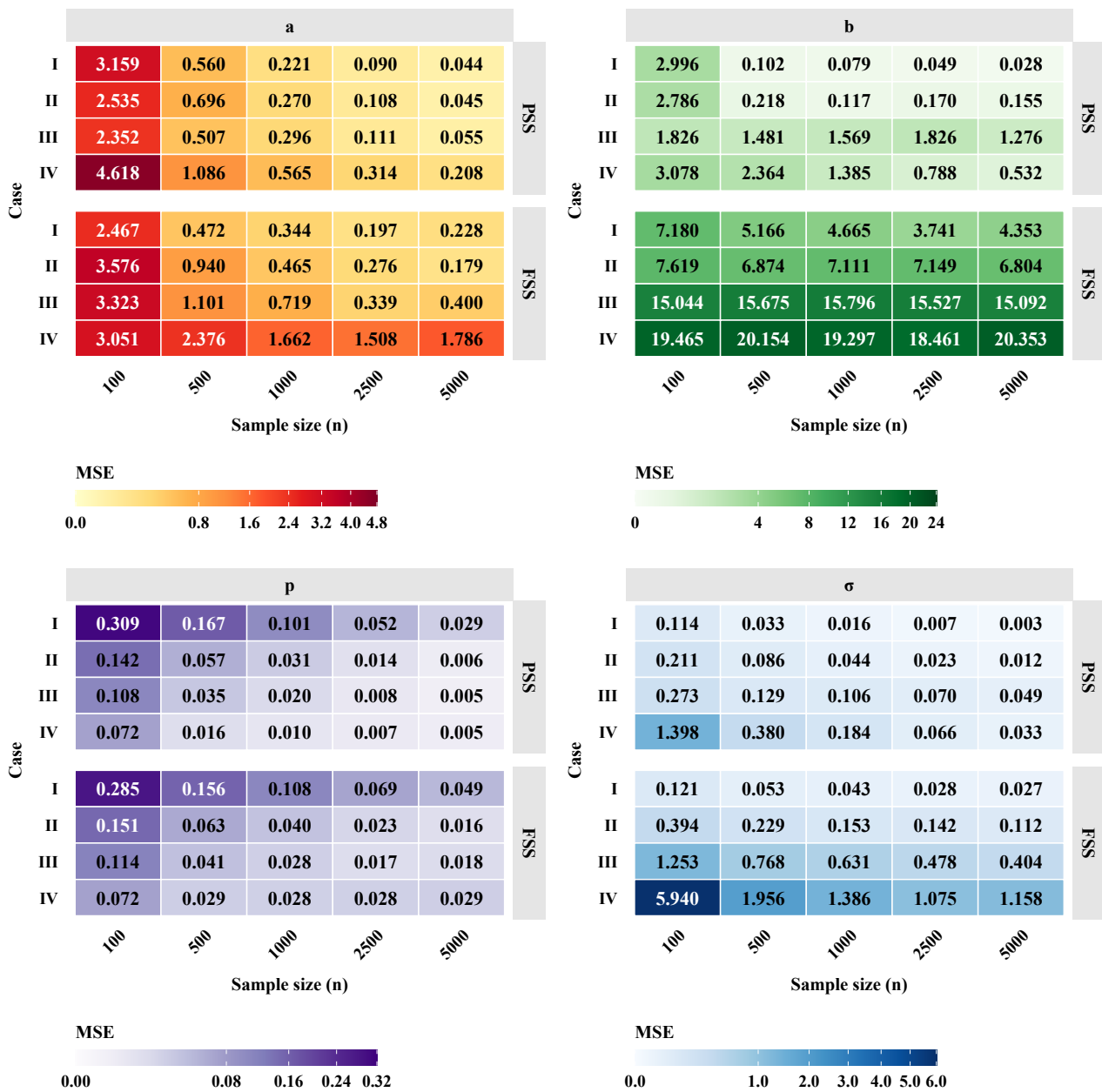


Figure 4. MSE heatmaps for PSO under the PSS and FSS.

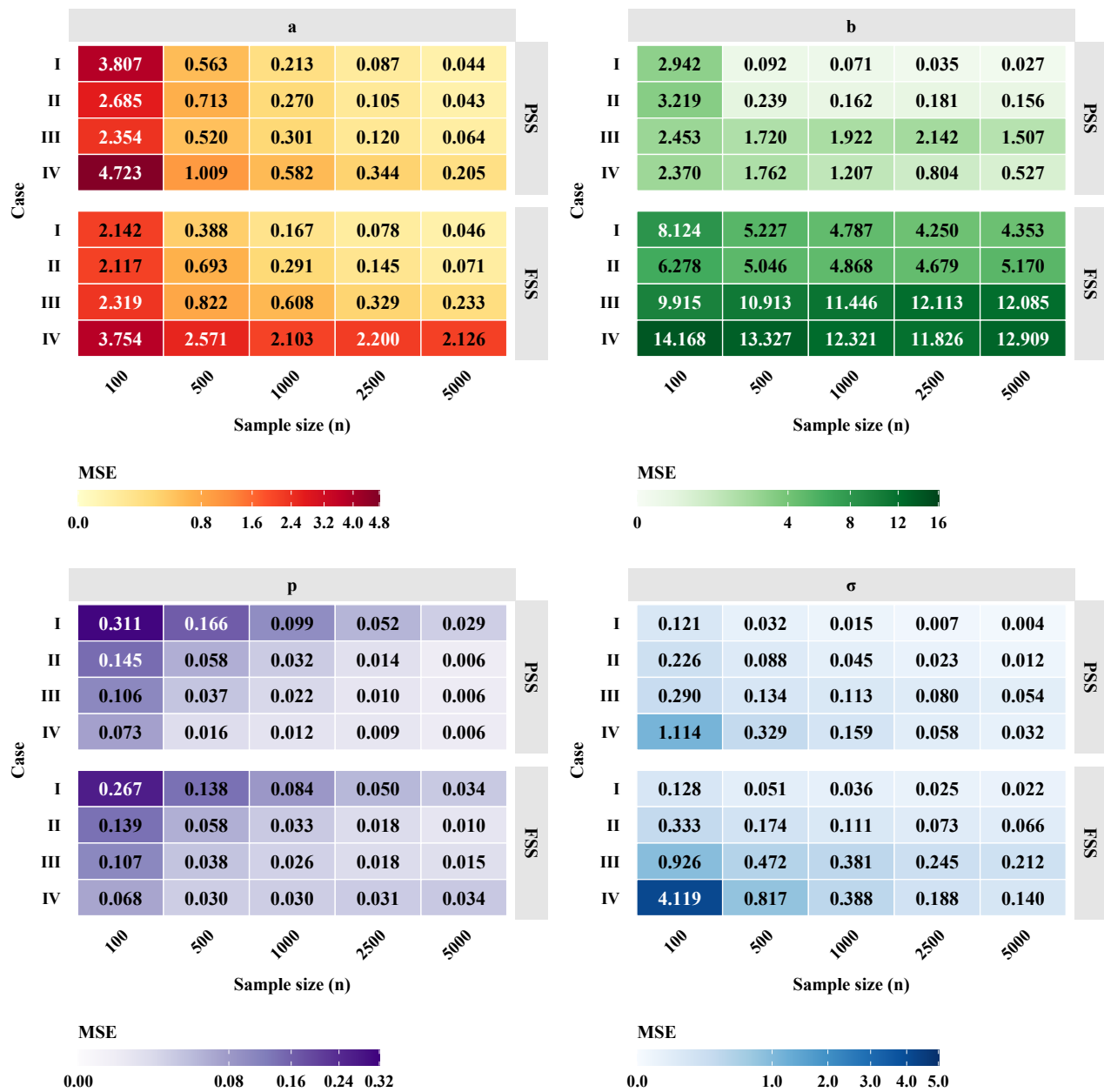


Figure 5. MSE heatmaps for DE under the PSS and FSS.



Figure 6. Heatmaps for Def values for GA, PSO, and DE under the PSS and FSS.

4.5. Additional Monte Carlo results

In addition to the main simulation results, additional analyses are conducted under the same parameter settings and sample sizes as in the main simulation study, using 1000 Monte Carlo replications for each scenario and 1000 bootstrap replications within each Monte Carlo replication. First, the empirical coverage (EC) and average width (AW) of the proposed intervals are examined to provide further insight into the behavior of the PSS.

The results demonstrate that the EC probabilities are generally close to the nominal level of 0.95, although some departures are observed across the simulation settings. In particular, some values above the nominal level suggest conservative coverage. However, this does not constitute a disadvantage for the intended use of these intervals because they are used to define data-driven search spaces rather than for inferential conclusions. The AW values generally decrease as the sample size increases, indicating progressively narrower search spaces, except for parameter b in Case III, for which the pattern is not monotonic. Overall, the results indicate that the bootstrap intervals generally become narrower as the sample size increases while maintaining EC close to the nominal level. See Table A1 in the Appendix for the corresponding EC and AW values.

Second, to assess the uncertainty arising from the finite number of Monte Carlo replications for the PSS and FSS-based algorithms, the Monte Carlo standard errors (MCSEs) of bias and MSE are calculated as

$$MCSE(Bias) = \sqrt{\frac{1}{s(s-1)} \sum_{i=1}^s (\hat{\theta}_i - \bar{\theta})^2}, \quad (4.4)$$

$$MCSE(MSE) = \sqrt{\frac{1}{s(s-1)} \sum_{i=1}^s \left[(\hat{\theta}_i - \bar{\theta})^2 - MSE \right]^2}, \quad (4.5)$$

respectively, where $\bar{\theta} = s^{-1} \sum_{i=1}^s \hat{\theta}_i$. Smaller MCSEs indicate greater Monte Carlo precision; see [47].

For the present study, the results reveal that the MCSE values for bias and MSE exhibit broadly similar patterns. Under the PSS, the MCSEs generally decrease with increasing sample size and are typically smaller than their FSS counterparts for larger samples. The reduction is most apparent for parameter b , particularly for MSE, and less pronounced for σ . For a and p , PSS generally yields smaller MCSEs as n increases. These MCSEs quantify the Monte Carlo uncertainty in the reported bias and MSE estimates and are not used as performance criteria for ranking the estimation methods. See Figures A1 and A2 in the Appendix for the corresponding MCSEs.

Finally, the computational performance of PSS and FSS-based GA, PSO, and DE algorithms is evaluated in terms of mean computation time and the average number of iterations required to reach convergence. All algorithms are implemented under equal optimization budgets and the same stopping criterion. For each run, the convergence iteration is defined as the first iteration after which the best objective value does not improve by more than a relative tolerance of 10^{-6} .

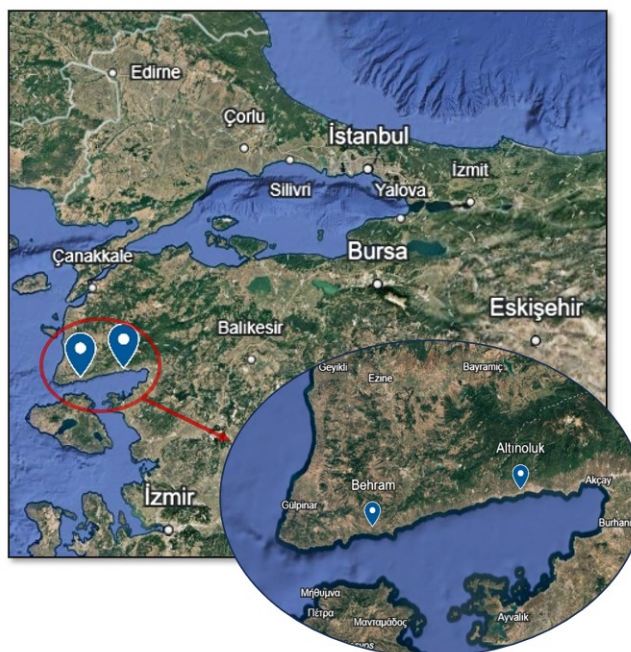
The average number of iterations required to reach convergence is less than 100 in every considered setting, suggesting that $\text{maxit} = 100$ provides a generally adequate iteration budget. Although convergence behavior varies across algorithms and sample sizes, PSS-based algorithms do not systematically require more iterations to reach convergence than their FSS-based counterparts. As expected, the reported mean computation times are longer under the PSS because they include both bootstrap-based search-space construction and algorithm execution, whereas the reported FSS times include only algorithm execution. However, results regarding the bias, MSE, and Def values indicate that PSS-based algorithms generally provide better estimation performance than their FSS-based counterparts in most of the considered scenarios. These improvements are achieved without systematically requiring more iterations. Therefore, the results indicate a reasonable trade-off between estimation performance and computational cost. See also Table A2 in the Appendix for more information.

5. Application

In this section, two wind speed datasets obtained from the Turkish State Meteorological stations named as Edremit/Altınoluk and Ayvacık/Behram, located in the northern part of the Edremit Gulf, Turkey, are analyzed to assess the modeling performance of the KumW distribution. The datasets consist of hourly mean wind speed measurements recorded at 10 m height throughout 2024. For the sake of brevity, these stations will hereafter be referred to as Altınoluk and Behram, respectively. The main reason for selecting these stations is the significant wind energy potential of the Edremit Gulf. Table 4 summarizes the geographical characteristics of the selected stations; see also Figure 7 for their geographical locations.

Table 4. Geographical characteristics of the Altınoluk and Behram stations.

Station	Latitude	Longitude	Elevation
Altınoluk	39°34'10.6''	26°42'45.0''	150 m
Behram	39°29'52.0''	26°18'51.0''	234 m

**Figure 7.** Locations of the Altınoluk and Behram stations.

Descriptive statistics for the wind speed datasets obtained from the above-mentioned stations are presented in Table 5.

Table 5. Descriptive statistics for the annual and seasonal wind speed datasets obtained from the Altınoluk and Behram stations.

	Mean	Variance	Skewness	Kurtosis	Max	<i>n</i>
Altınoluk						
Annual	2.0297	1.4751	1.5173	7.1124	14.0	8596
Winter	2.0298	1.5536	1.931	10.8303	14.0	2135
Spring	1.9555	1.294	1.3606	5.8236	8.5	2144
Summer	2.1372	1.6443	1.2061	5.0736	9.4	2169
Autumn	1.9952	1.3907	1.5218	6.2426	8.3	2148
Behram						
Annual	4.1031	6.9232	1.1396	4.3153	18.3	8627
Winter	4.4415	9.1288	1.2289	4.4523	18.3	2147
Spring	3.7547	5.905	1.165	4.3585	14.5	2175
Summer	4.4954	7.3391	0.7698	2.9274	16.1	2171
Autumn	3.7185	4.785	1.0462	4.2016	13.9	2134

First, it should be noted that the original datasets contain 8784 hourly observations for each station. Prior to distribution fitting and parameter estimation, zero-valued observations are excluded from the wind speed datasets for the following reasons: They may actually arise from measurement precision and denote calm conditions rather than indicating true zero wind speeds, and they account for less than 2% of the observations in the original datasets. The exclusion of zero-valued observations from wind speed analysis is also consistent with practices reported in the literature; see, for example, [48,49]. In this study, 25 observations are not available, and 163 zero-valued observations (1.86%) are excluded, leaving 8596 observations for the Altınoluk station, whereas 88 observations are not available, and 69 zero-valued observations (0.79%) are excluded, leaving 8627 observations for the Behram station.

The wind speed behavior at the Altınoluk and Behram stations shows notable differences across different times of the year. Behram experiences higher average wind speeds throughout the year, with an annual mean speed of approximately 4.10 m/s, nearly twice that of Altınoluk, which records an annual mean of 2.03 m/s. In Behram, the highest seasonal wind speeds occur in summer and winter, whereas spring and autumn exhibit comparatively lower values. On the other hand, in Altınoluk, summer is the windiest among the seasons, whereas spring has the lowest average. Seasonal fluctuations in wind speeds are greater in Behram, as reflected by its higher variances, whereas Altınoluk exhibits relatively stable wind behavior throughout the year. The wind speed data from both stations show positive skewness, with Altınoluk having higher values. All kurtosis values are above 3 except for summer in Behram, indicating that the wind speed distributions are leptokurtic. Behram records a maximum wind speed of 18.3 m/s, and the highest value in Altınoluk is 14.0 m/s. Overall, the results suggest that both regions appear to have potential in terms of wind energy.

Here, our aim is to obtain the parameter estimates for the KumW distribution, providing the best fitting performance for modeling the wind speed datasets obtained from the Altınoluk and Behram stations. Another objective is to compare the fitting performance of the KumW distribution with that of the Weibull distribution, a widely used model for wind speed analysis and a special case of the KumW distribution, as noted earlier.

It should be noted that the hourly wind speed observations may be serially correlated. For example, Ramirez and Carta [50] reported that, although autocorrelation may have little effect on the shape of the fitted wind speed distribution, the uncertainty measures and confidence intervals obtained from dependent wind speed data may be overoptimistic. Therefore, the dependence structure of the annual and seasonal datasets is examined in this study, and serial dependence is detected. To account for this dependence when constructing the PSS, the ordinary bootstrap is replaced by the moving block bootstrap (MBB). The resulting bootstrap intervals are then used to define the PSS.

The comparisons are performed using the following statistical criteria:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{F}_i - F_i)^2}, \quad (5.1)$$

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |\hat{F}_i - F_i|, \quad (5.2)$$

$$R^2 = 1 - \frac{\sum_{i=1}^n (\hat{F}_i - F_i)^2}{\sum_{i=1}^n (F_i - \bar{F})^2}, \quad (5.3)$$

$$\text{AIC} = -2 \ln L + 2k, \quad (5.4)$$

$$\text{KS} = \max \{D_n^+, D_n^-\}, \quad (5.5)$$

and

$$\text{PDE} = \left| \frac{P_{ref} - P_D}{P_{ref}} \right|. \quad (5.6)$$

Here, $F_i = i/(n+1)$, $\hat{F}_i = \hat{F}(x_{(i)})$, and $\bar{F} = (1/n) \sum_{i=1}^n F_i$, where $x_{(i)}$ denotes the i th ordered observation. Furthermore, L is the maximized likelihood value for the fitted distribution, and k is the number of estimated parameters. For the KS criterion, D_n^+ and D_n^- are calculated as $D_n^+ = \max_{1 \leq i \leq n} \{(i/n) - \hat{F}_i\}$ and $D_n^- = \max_{1 \leq i \leq n} \{\hat{F}_i - ((i-1)/n)\}$, respectively. P_{ref} is the mean wind power density based on observed data, and P_D is the mean wind power density based on the distributional model. The equalities P_{ref} and P_D are given below:

$$P_{ref} = \frac{1}{2} \rho \frac{1}{n} \sum_{i=1}^n x_i^3 \quad \text{and} \quad P_D = \frac{1}{2} \rho \int_0^\infty x^3 f_D(x) dx,$$

respectively. Here, ρ is the air density (kg/m^3), and f_D refers to the PDF of the distribution of interest, that is, the KumW or Weibull distribution. Note that upon substituting P_{ref} and P_D into (5.6), the common factor $(1/2)\rho$ is eliminated through algebraic simplification. It should be noticed that lower RMSE, MAE, AIC, KS, and PDE values and higher R^2 values are associated with a better fit.

First, the ML estimates of the parameters of KumW and Weibull distributions are obtained using GA(PSS); see Table 6. The primary reason for utilizing GA(PSS) in the estimation stage is its superior performance compared to other methods; see the Monte Carlo simulation results given in Section 4.

Then, RMSE, MAE, R^2 , AIC, KS, and PDE values are computed for both KumW and Weibull distributions and reported in Table 7. Table 7 indicates that the KumW distribution generally provides a better fit than the Weibull distribution across the annual and seasonal datasets. Specifically, KumW has lower RMSE, MAE, and AIC values and higher R^2 values than Weibull for all datasets at both stations. The KS values are also lower for KumW in all datasets except for winter at Behram, where Weibull has a slightly lower KS value. At Altinoluk, KumW has lower PDE values except in summer, where Weibull has a slightly lower value. At Behram, KumW has a lower PDE value only in spring, whereas Weibull has lower values for the remaining datasets.

The histograms for the annual and seasonal wind speed data obtained from the Altinoluk and Behram stations and the corresponding fitted curves for the KumW and Weibull distributions are given in Figures 8 and 9.

Results drawn from Figures 8 and 9 are in agreement with the findings obtained from Table 7. See also the following details regarding Figures 8 and 9. KumW distribution models the peakness of the wind speed data better than the Weibull distribution for all cases. When modeling the tails, KumW distribution is superior to the Weibull distribution for the wind speed data from the Altinoluk station except for the summer season, where both the KumW and Weibull distributions show similar performances; see Figure 8. Their performances are also comparable in modeling the tails of the distribution for the wind speed data from the Behram station; see Figure 9.

Overall, the results given in this section show that the KumW distribution can be considered as a strong alternative to the Weibull distribution for modeling wind speed data.

Table 6. The estimates of the unknown parameters of KumW and Weibull distributions for the annual and seasonal wind speed datasets obtained from the Altınoluk and Behram stations.

		Distribution	$\hat{a}$	$\hat{b}$	$\hat{p}$	$\hat{\sigma}$
Altınoluk	Annual	KumW	5.9294	2.1400	0.6903	0.8635
		Weibull	–	–	1.9364	2.2328
	Winter	KumW	5.8801	3.1596	0.6423	1.0270
		Weibull	–	–	1.9338	2.2619
	Spring	KumW	3.2869	3.8455	0.8448	2.0780
		Weibull	–	–	1.9501	2.2036
	Summer	KumW	5.6800	3.9075	0.6071	1.2100
		Weibull	–	–	1.7604	2.4079
	Autumn	KumW	7.6240	2.0106	0.6505	0.6446
		Weibull	–	–	1.9791	2.2187
Behram	Annual	KumW	2.9367	1.0863	0.9694	2.3244
		Weibull	–	–	1.6504	4.6095
	Winter	KumW	3.6314	0.9286	0.8740	1.8803
		Weibull	–	–	1.5626	4.9695
	Spring	KumW	3.4983	3.1002	0.7509	3.2197
		Weibull	–	–	1.6341	4.2140
	Summer	KumW	2.7811	5.4349	0.8304	6.9329
		Weibull	–	–	1.7185	5.0473
	Autumn	KumW	3.5991	3.1089	0.8097	3.2132
		Weibull	–	–	1.7999	4.1975

Table 7. RMSE, MAE, R^2 , AIC, KS, and PDE values for the wind speed distributions.

		Distribution	RMSE	MAE	R^2	AIC	KS	PDE
Altınoluk	Annual	KumW	0.0139	0.0106	0.9977	25228.89	0.0381	0.0219
		Weibull	0.0240	0.0208	0.9931	25748.77	0.0512	0.2360
	Winter	KumW	0.0173	0.0138	0.9964	6237.78	0.0426	0.0710
		Weibull	0.0309	0.0261	0.9885	6424.31	0.0679	0.2596
	Spring	KumW	0.0174	0.0139	0.9964	6106.69	0.0417	0.0312
		Weibull	0.0265	0.0216	0.9916	6194.10	0.0625	0.1434
	Summer	KumW	0.0116	0.0095	0.9984	6685.96	0.0328	0.0564
		Weibull	0.0247	0.0200	0.9927	6725.25	0.0583	0.0526
	Autumn	KumW	0.0167	0.0132	0.9966	6143.25	0.0426	0.0186
		Weibull	0.0328	0.0277	0.9871	6312.06	0.0719	0.2172
Behram	Annual	KumW	0.0113	0.0096	0.9985	38457.81	0.0287	0.0595
		Weibull	0.0212	0.0182	0.9946	38701.98	0.0408	0.0479
	Winter	KumW	0.0162	0.0127	0.9968	10022.03	0.0430	0.1286
		Weibull	0.0229	0.0204	0.9937	10106.24	0.0424	0.0609
	Spring	KumW	0.0109	0.0092	0.9986	9347.72	0.0290	0.0291
		Weibull	0.0199	0.0172	0.9953	9399.78	0.0411	0.0499
	Summer	KumW	0.0187	0.0160	0.9958	10026.61	0.0378	0.0622
		Weibull	0.0211	0.0180	0.9947	10028.52	0.0458	0.0049
	Autumn	KumW	0.0109	0.0090	0.9986	8875.66	0.0334	0.0414
		Weibull	0.0200	0.0170	0.9952	8925.46	0.0421	0.0383

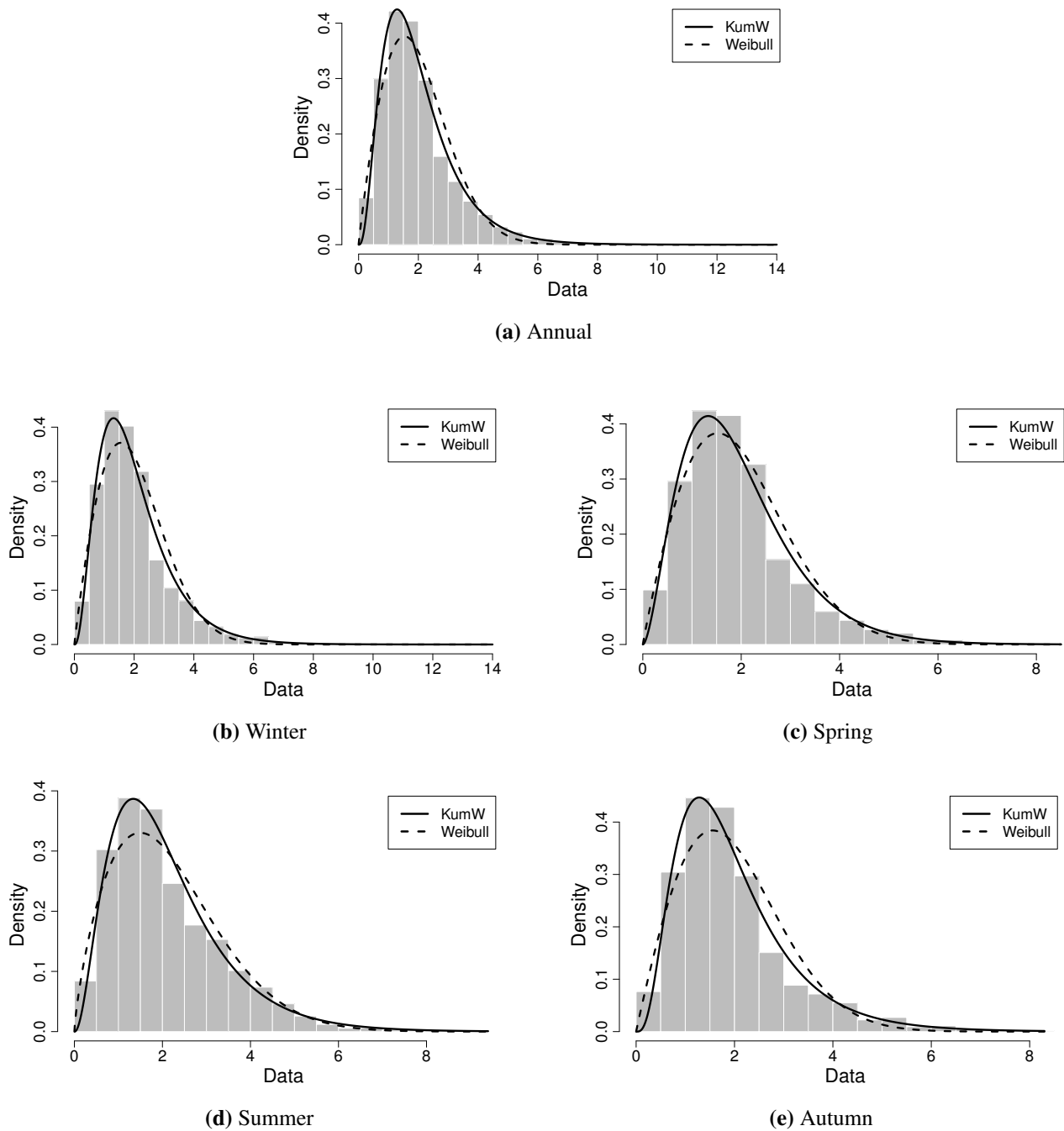


Figure 8. The fitted density plots and the histograms for the annual and seasonal wind speed data obtained from the Altınoluk station.

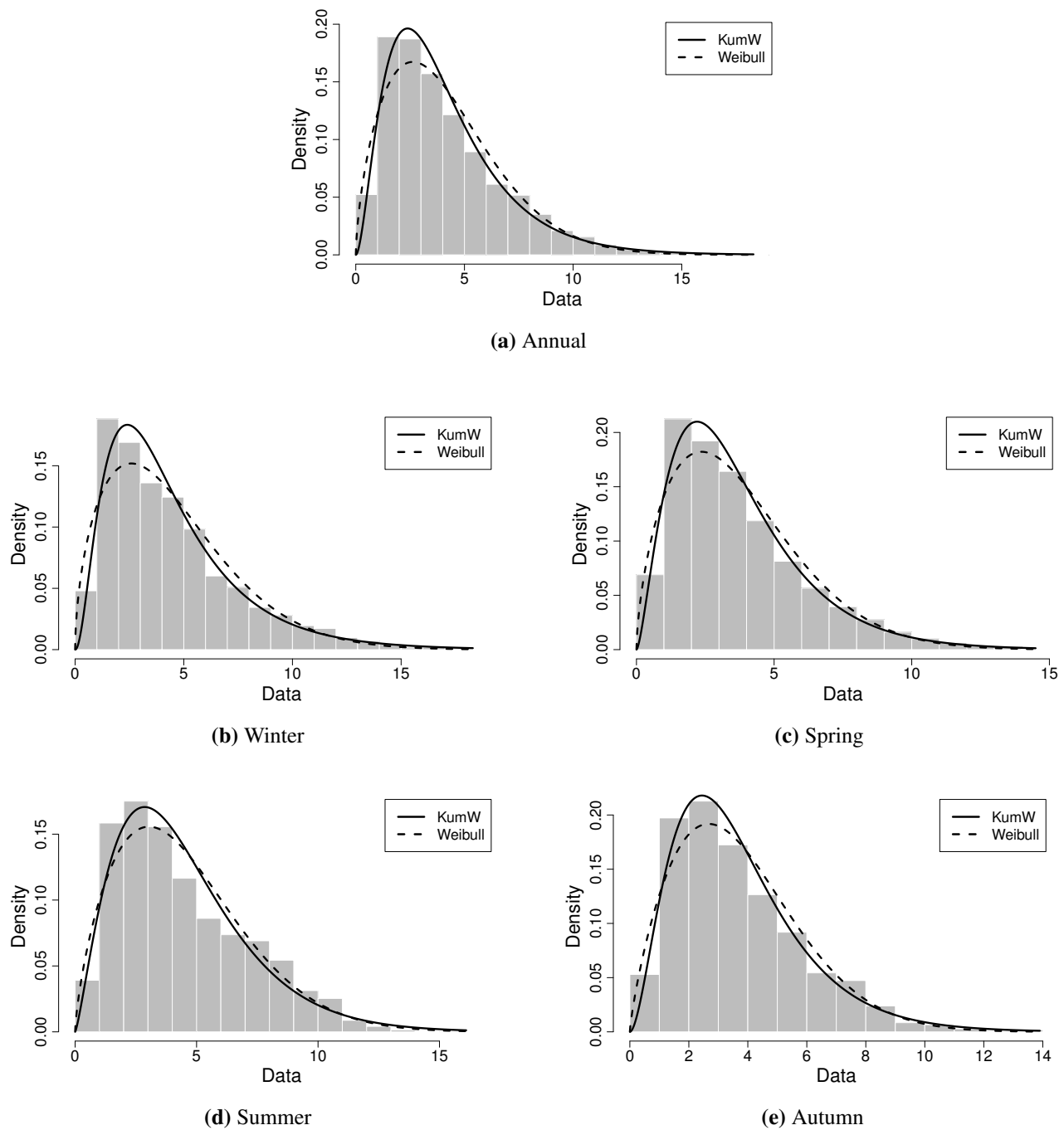


Figure 9. The fitted density plots and the histograms for the annual and seasonal wind speed data obtained from the Behram station.

6. Conclusions

In this study, the KumW distribution, a flexible generalization of the widely used Weibull distribution, is applied to model wind speed data. In estimating the unknown KumW parameters, the ML methodology is utilized; however, the corresponding likelihood equations do not have closed form solutions. To overcome this problem, three popular metaheuristic algorithms, GA, PSO, and DE, are employed instead of traditional numerical methods, which often involve computational difficulties. A novel data-driven search space, based on the bootstrap CIs of the KumW parameters, is proposed to enhance the performance of the GA, PSO, and DE algorithms.

The effect of using the proposed search space, referred to as the PSS, instead of its conventional counterpart, the FSS, on the performances of GA, PSO, and DE is evaluated through a Monte Carlo simulation study. The simulation results indicate that incorporating the PSS into the metaheuristic algorithms considered in this study as the search space significantly enhances their performance in estimating the parameters of the KumW distribution. This advantage becomes more pronounced when the sample size increases, and the underlying distribution becomes more skewed and leptokurtic. Among the metaheuristic algorithms, GA demonstrates superior performance compared to the others, particularly in the case of highly skewed and heavy-tailed distributions with relatively small sample sizes.

In the application part of the study, the wind speed data obtained from the Altinoluk and Behram meteorological stations located in the northern part of the Edremit Gulf, Turkey are analyzed. For the estimation of the parameters of the KumW distribution and its widely used submodel in the literature, the Weibull distribution, GA(PSS), which demonstrated the best performance according to the simulation results, is employed. Using the parameter estimates obtained separately for annual and seasonal wind speed data, the relevant datasets are modeled. The results showed that the KumW distribution provides a better fit to the data than the widely used Weibull distribution in nearly all of the examined cases. It should also be noted that there exist various studies focusing on the joint modeling of wind speed with other meteorological variables, such as wind direction, air temperature, and air density. However, it is obvious that the present analysis is limited to modeling the marginal distribution of wind speed.

Overall, GA, PSO, and DE algorithms based on the PSS navigate the solution space more efficiently and therefore provide more accurate estimates than their conventional counterparts based on the FSS. As a result, the use of the PSS rather than the FSS as the search space in the metaheuristic algorithms considered in this study is recommended in order to enhance the estimation accuracy of the unknown KumW parameters. This study also highlights that the KumW distribution, a generalization of the Weibull, can reliably be used to model the wind speed data.

In our future studies, we are planning to extend the proposed method to other probability distributions. Moreover, alternative resampling methods for determining the search space will be investigated.

Use of AI tools declaration

During the preparation of this work, the authors used ChatGPT (GPT-5) in order to check the grammatical correctness of some sentences in the manuscript. After using this tool, the authors

reviewed and edited the content as needed, and they take full responsibility for the content of the published article.

Conflict of interest

The authors declare there are no conflicts of interest.

References

1. S. Huang, Q. Li, Z. Shu, P. W. Chan, Copula-based estimation of directional extreme wind speeds: Application for wind-resistant structural design, *Structures*, **60** (2024), 105845. <https://doi.org/10.1016/j.istruc.2023.105845>
2. Y. Feng, M. G. Stewart, Probabilistic analysis of wind-induced failures of transmission tower-line systems, *Struct. Infrastruct. Eng.*, **21** (2025), 2101–2114. <https://doi.org/10.1080/15732479.2025.2554724>
3. Y. Wang, T. Deng, J. Fu, M. Du, X. Hu, Monte Carlo-based full-probability analysis of wind-induced responses in super-tall buildings considering the joint distribution of wind speed and direction, *J. Build. Eng.*, **112** (2025), 113646. <https://doi.org/10.1016/j.jobbe.2025.113646>
4. Y. Wang, T. Deng, J. Fu, Investigation of wind load characteristics and structural responses in asymmetric linked twin-tower high-rise buildings, *Eng. Struct.*, **341** (2025), 120796. <https://doi.org/10.1016/j.engstruct.2025.120796>
5. S. E. Tuller, A. C. Brett, The characteristics of wind velocity that favor the fitting of a weibull distribution in wind speed analysis, *J. Appl. Meteorol. Climatol.*, **23** (1984), 124–134. [https://doi.org/10.1175/1520-0450\(1984\)023<0124:TCOWVT>2.0.CO;2](https://doi.org/10.1175/1520-0450(1984)023<0124:TCOWVT>2.0.CO;2)
6. J. Seguro, T. Lambert, Modern estimation of the parameters of the weibull wind speed distribution for wind energy analysis, *J. Wind Eng. Ind. Aerodyn.*, **85** (2000), 75–84. [https://doi.org/10.1016/S0167-6105\(99\)00122-1](https://doi.org/10.1016/S0167-6105(99)00122-1)
7. M. Soulouknga, S. Doka, N. Revanna, N. Djongyang, T. Kofane, Analysis of wind speed data and wind energy potential in faya-largeau, chad, using Weibull distribution, *Renewable Energy*, **121** (2018), 1–8. <https://doi.org/10.1016/j.renene.2018.01.002>
8. W. Wang, T. Okaze, Statistical analysis of low-occurrence strong wind speeds at the pedestrian level around a simplified building based on the Weibull distribution, *Build. Environ.*, **209** (2022), 108644. <https://doi.org/10.1016/j.buildenv.2021.108644>
9. O. Jaramillo, M. Borja, Wind speed analysis in la ventosa, mexico: a bimodal probability distribution case, *Renewable Energy*, **29** (2004), 1613–1630. <https://doi.org/10.1016/j.renene.2004.02.001>
10. S. Akpinar, E. K. Akpinar, Wind energy analysis based on maximum entropy principle (MEP)-type distribution function, *Energy Convers. Manage.*, **48** (2007), 1140–1149. <https://doi.org/10.1016/j.enconman.2006.10.004>
11. V. L. Brano, A. Orioli, G. Ciulla, S. Culotta, Quality of wind speed fitting distributions for the urban area of Palermo, Italy, *Renewable Energy*, **36** (2011), 1026–1039. <https://doi.org/10.1016/j.renene.2010.09.009>

12. K. Philippopoulos, D. Deligiorgi, G. Karvounis, Wind speed distribution modeling in the greater area of Chania, Greece, *Int. J. Green Energy*, **9** (2012), 174–193. <https://doi.org/10.1080/15435075.2011.622020>
13. P. Drobinski, C. Coulais, B. Jourdier, Surface wind-speed statistics modelling: Alternatives to the Weibull distribution and performance evaluation, *Boundary Layer Meteorol.*, **157** (2015), 97–123. <https://doi.org/10.1007/s10546-015-0035-7>
14. F. G. Akgül, B. Şenoğlu, T. Arslan, An alternative distribution to Weibull for modeling the wind speed data: inverse Weibull distribution, *Energy Convers. Manage.*, **114** (2016), 234–240. <https://doi.org/10.1016/j.enconman.2016.02.026>
15. O. Alavi, K. Mohammadi, A. Mostafaeipour, Evaluating the suitability of wind speed probability distribution models: A case of study of east and southeast parts of Iran, *Energy Convers. Manage.*, **119** (2016), 101–108. <https://doi.org/10.1016/j.enconman.2016.04.039>
16. E. Chiodo, P. D. Falco, Inverse Burr distribution for extreme wind speed prediction: Genesis, identification and estimation, *Electr. Power Syst. Res.*, **141** (2016), 549–561. <https://doi.org/10.1016/j.epsr.2016.08.028>
17. T. Arslan, Ş. Acitaş, B. Şenoğlu, Generalized Lindley and Power Lindley distributions for modeling the wind speed data, *Energy Convers. Manage.*, **152** (2017), 300–311. <https://doi.org/10.1016/j.enconman.2017.08.017>
18. N. Aries, S. M. Boudia, H. Ounis, Deep assessment of wind speed distribution models: A case study of four sites in Algeria, *Energy Convers. Manage.*, **155** (2018), 78–90. <https://doi.org/10.1016/j.enconman.2017.10.082>
19. K. Bagci, T. Arslan, H. E. Celik, Inverted Kumaraswamy distribution for modeling the wind speed data: Lake Van, Turkey, *Renewable Sustainable Energy Rev.*, **135** (2021), 110110. <https://doi.org/10.1016/j.rser.2020.110110>
20. M. A. ul Haq, S. Hashmi, M. Aslam, Marshall-Olkin length biased exponential distribution for wind speed analysis alternative to Weibull distribution, *Model. Earth Syst. Environ.*, **10** (2024), 1095–1108. <https://doi.org/10.1007/s40808-023-01832-0>
21. G. M. Cordeiro, E. M. Ortega, S. Nadarajah, The Kumaraswamy Weibull distribution with application to failure data, *J. Franklin Inst.*, **347** (2010), 1399–1429. <https://doi.org/10.1016/j.jfranklin.2010.06.010>
22. R. Brits, A. P. Engelbrecht, F. van den Bergh, Locating multiple optima using particle swarm optimization, *Appl. Math. Comput.*, **189** (2007), 1859–1883. <https://doi.org/10.1016/j.amc.2006.12.066>
23. G. Armano, M. R. Farmani, Multiobjective clustering analysis using particle swarm optimization, *Expert Syst. Appl.*, **55** (2016), 184–193. <https://doi.org/10.1016/j.eswa.2016.02.009>
24. C. K. Ting, X. L. Liao, Y. H. Huang, R. T. Liaw, Multi-vehicle selective pickup and delivery using metaheuristic algorithms, *Inf. Sci.*, **406** (2017), 146–169. <https://doi.org/10.1016/j.ins.2017.04.001>

25. A. Yalçınkaya, B. Şenoğlu, U. Yolcu, Maximum likelihood estimation for the parameters of skew normal distribution using genetic algorithm, *Swarm Evol. Comput.*, **38** (2018), 127–138. <https://doi.org/10.1016/j.swevo.2017.07.007>
26. S. Acitas, C. H. Aladag, B. Senoglu, A new approach for estimating the parameters of Weibull distribution via particle swarm optimization: An application to the strengths of glass fibre data, *Reliab. Eng. Syst. Saf.*, **183** (2019), 116–127. <https://doi.org/10.1016/j.res.2018.07.024>
27. S. Pech, G. Kandler, M. Lukacevic, J. Füssl, Metamodel assisted optimization of glued laminated timber beams by using metaheuristic algorithms, *Eng. Appl. Artif. Intell.*, **79** (2019), 129–141. <https://doi.org/10.1016/j.engappai.2018.12.010>
28. Z. Stokes, A. Mandal, W. K. Wong, Using Differential Evolution to design optimal experiments, *Chemom. Intell. Lab. Syst.*, **199** (2020), 103955. <https://doi.org/10.1016/j.chemolab.2020.103955>
29. C. Huang, Y. Bai, Y. Yan, Q. Zhang, N. Zhang, W. Wang, Multi-objective co-optimization of design and operation in an independent solar-based distributed energy system using genetic algorithm, *Energy Convers. Manage.*, **271** (2022), 116283. <https://doi.org/10.1016/j.enconman.2022.116283>
30. T. P. Chang, Wind energy assessment incorporating particle swarm optimization method, *Energy Convers. Manage.*, **52** (2011), 1630–1637. <https://doi.org/10.1016/j.enconman.2010.10.024>
31. F. J. Liu, P. H. Chen, S. S. Kuo, D. C. Su, T. P. Chang, Y. H. Yu, et al., Wind characterization analysis incorporating genetic algorithm: A case study in Taiwan Strait, *Energy*, **36** (2011), 2611–2619. <https://doi.org/10.1016/j.energy.2011.02.001>
32. M. Lydia, S. S. Kumar, A. I. Selvakumar, G. E. P. Kumar, Wind resource estimation using wind speed and power curve models, *Renewable Energy*, **83** (2015), 425–434. <https://doi.org/10.1016/j.renene.2015.04.045>
33. T. C. Carneiro, S. P. Melo, P. C. Carvalho, A. P. d. S. Braga, Particle Swarm Optimization method for estimation of Weibull parameters: A case study for the Brazilian northeast region, *Renewable Energy*, **86** (2016), 751–759. <https://doi.org/10.1016/j.renene.2015.08.060>
34. P. Krömer, J. Heckenbergerová, P. Musilek, Accurate mixed weibull distribution fitting by differential evolution, in *GECCO '17: Proceedings of the Genetic and Evolutionary Computation Conference*, (2017), 1161–1168. <https://doi.org/10.1145/3071178.3071290>
35. R. Kumar, A. Kumar, Application of differential evolution for wind speed distribution parameters estimation, *Wind Eng.*, **45** (2021), 1544–1556. <https://doi.org/10.1177/0309524X21999964>
36. A. Amirjanov, The development of a changing range genetic algorithm, *Comput. Methods Appl. Mech. Eng.*, **195** (2006), 2495–2508. <https://doi.org/10.1016/j.cma.2005.05.014>
37. A. O. Adewumi, A. M. Arasomwan, Improved particle swarm optimizer with dynamically adjusted search space and velocity limits for global optimization, *Int. J. Artif. Intell. Tools*, **24** (2015), 1550017. <https://doi.org/10.1142/S0218213015500177>
38. L. Luo, X. Hou, J. Zhong, W. Cai, J. Ma, Sampling-based adaptive bounding evolutionary algorithm for continuous optimization problems, *Inf. Sci.*, **382** (2017), 216–233. <https://doi.org/10.1016/j.ins.2016.12.023>

39. O. Ouédraogo, E. Katchekpele, T. A. Kpanzou, Adaptive bounds and estimation in the case of the three-parameter Weibull distribution, *Afr. Stat.*, **16** (2021), 2979–2991. <https://doi.org/10.16929/as/2021.2979.191>
40. J. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, The MIT Press, 1992. <https://doi.org/10.7551/mitpress/1090.001.0001>
41. J. Kennedy, R. Eberhart, Particle swarm optimization, in *Proceedings of ICNN'95 - International Conference on Neural Networks*, **4** (1995), 1942–1948. <https://doi.org/10.1109/ICNN.1995.488968>
42. R. Storn, K. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, **11** (1997), 341–359. <https://doi.org/10.1023/A:1008202821328>
43. M. Ahsan-ul-Haq, S. M. Choudhary, A. H. AL-Marshadi, M. Aslam, A new generalization of Lindley distribution for modeling of wind speed data, *Energy Rep.*, **8** (2022), 1–11. <https://doi.org/10.1016/j.egy.2021.11.246>
44. F. G. Akgül, B. Şenoğlu, Comparison of wind speed distributions: a case study for Aegean coast of Turkey, *Energy Sources Part A*, **45** (2023), 2453–2470. <https://doi.org/10.1080/15567036.2019.1663309>
45. I. Pobočková, M. Michalková, Z. Sedláčková, D. Jurášová, Modelling the wind speed using exponentiated Weibull distribution: Case study of Poprad-Tatry, Slovakia, *Appl. Sci.*, **13** (2023), 4031. <https://doi.org/10.3390/app13064031>
46. T. Yang, D. Chen, Beta-generalized Lindley distribution: A novel probability model for wind speed, *Renewable Energy*, **256** (2026), 124216. <https://doi.org/10.1016/j.renene.2025.124216>
47. T. P. Morris, I. R. White, M. J. Crowther, Using simulation studies to evaluate statistical methods, *Stat. Med.*, **38** (2019), 2074–2102. <https://doi.org/10.1002/sim.8086>
48. R. Exell, The wind energy potential of Thailand, *Solar Energy*, **35** (1985), 3–13. [https://doi.org/10.1016/0038-092X\(85\)90031-3](https://doi.org/10.1016/0038-092X(85)90031-3)
49. N. Nawri, G.N. Petersen, H. Bjornsson, A.N. Hahmann, K. Jónasson, C.B. Hasager, et al., The wind energy potential of Iceland, *Renewable Energy*, **69** (2014), 290–299. <https://doi.org/10.1016/j.renene.2014.03.040>
50. P. Ramírez, J. A. Carta, Influence of the data sampling interval in the estimation of the parameters of the Weibull wind speed probability density distribution: A case study, *Energy Convers. Manage.*, **46** (2005), 2419–2438. <https://doi.org/10.1016/j.enconman.2004.11.004>

Appendix

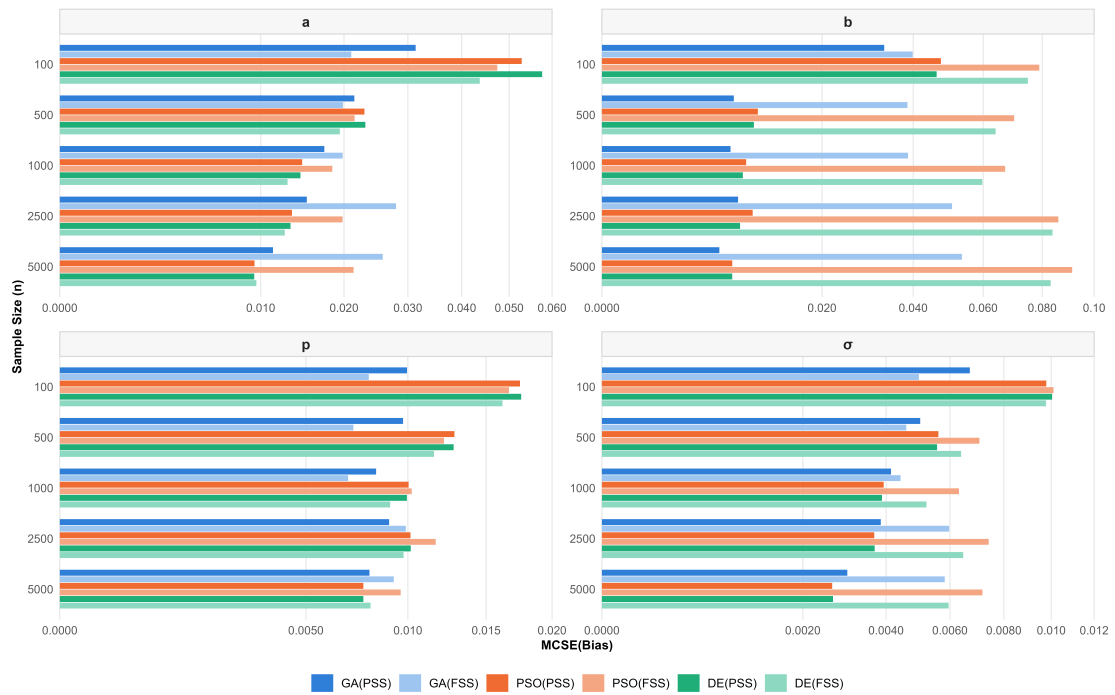
This section presents additional results supporting the main simulation study, including the empirical coverage and average width of the bootstrap intervals, the Monte Carlo standard errors of bias and MSE values, and the mean computation times and average numbers of iterations required for convergence for the PSS and FSS-based GA, PSO, and DE algorithms.

A.1. Empirical coverage and average width results

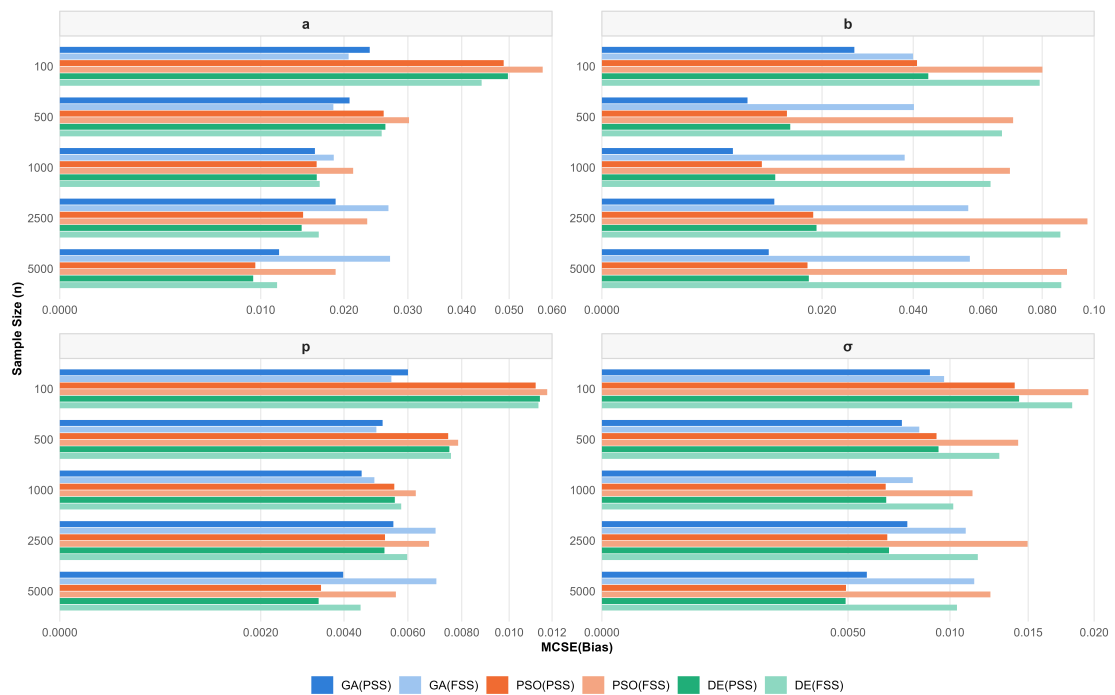
Table A1. Empirical coverage (EC) and average width (AW) of the proposed intervals.

n	a		b		p		σ	
	EC	AW	EC	AW	EC	AW	EC	AW
n	Case I							
100	0.9690	7.8077	0.9830	2.1516	0.9700	1.8736	0.9730	1.1335
500	0.9410	4.8539	0.9470	0.6599	0.9390	1.4338	0.9480	0.8046
1000	0.9590	3.1747	0.9470	0.5495	0.9600	1.2798	0.9550	0.6468
2500	0.9340	1.7947	0.9480	0.5096	0.9340	1.0277	0.9360	0.4383
5000	0.9480	1.2133	0.9720	0.4474	0.9460	0.8593	0.9500	0.3208
n	Case II							
100	0.9470	6.9026	0.9290	2.2250	0.9460	1.2144	0.9540	1.5423
500	0.9500	4.8748	0.9710	0.9259	0.9540	0.9588	0.9520	1.1894
1000	0.9530	3.3940	0.9670	0.8263	0.9570	0.8317	0.9520	1.0145
2500	0.9520	2.0426	0.9640	0.8129	0.9500	0.6481	0.9540	0.7753
5000	0.9420	1.3569	0.9900	0.7272	0.9460	0.4892	0.9400	0.5975
n	Case III							
100	0.9500	6.9056	0.9740	2.3676	0.9560	1.0853	0.9520	1.7485
500	0.9540	4.7156	0.9720	2.1293	0.9550	0.8204	0.9590	1.4287
1000	0.9400	3.3245	0.9710	2.2453	0.9450	0.6842	0.9380	1.2339
2500	0.9700	1.9396	0.9860	2.5140	0.9760	0.5014	0.9720	1.0097
5000	0.9580	1.3343	0.9860	2.2594	0.9700	0.3564	0.9600	0.7829
n	Case IV							
100	0.9560	8.0592	0.9630	3.8122	0.9650	0.9938	0.9570	3.5109
500	0.9590	5.3484	0.9670	3.5176	0.9710	0.6079	0.9580	2.7301
1000	0.9680	3.6167	0.9790	3.0917	0.9790	0.4495	0.9610	2.1373
2500	0.9600	2.3023	0.9600	2.5872	0.9880	0.3053	0.9600	1.3694
5000	0.9740	1.7435	0.9820	2.2313	0.9920	0.2554	0.9760	0.9513

A.2. Monte Carlo standard errors of bias and MSE values

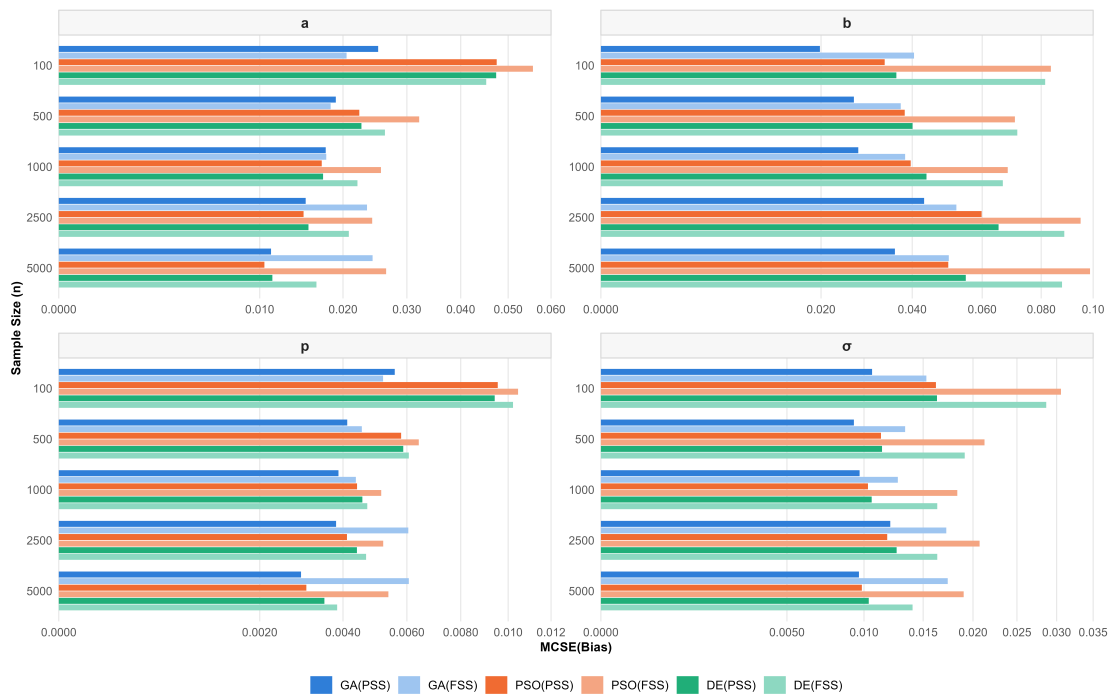


(a) Case I

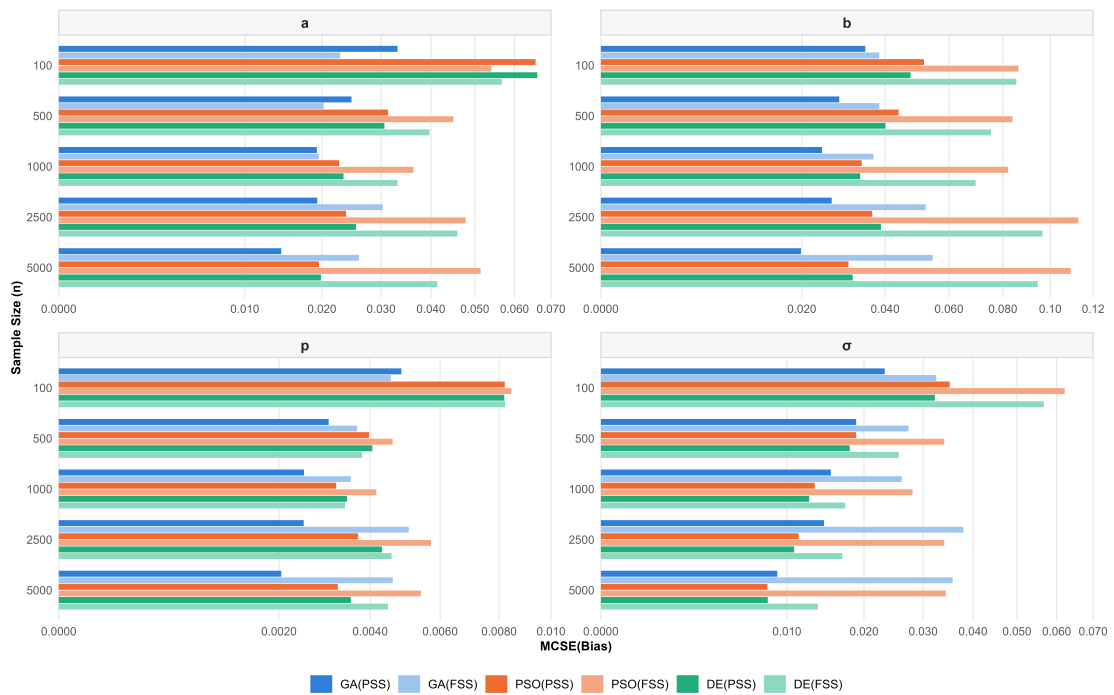


(b) Case II

Continued on next page

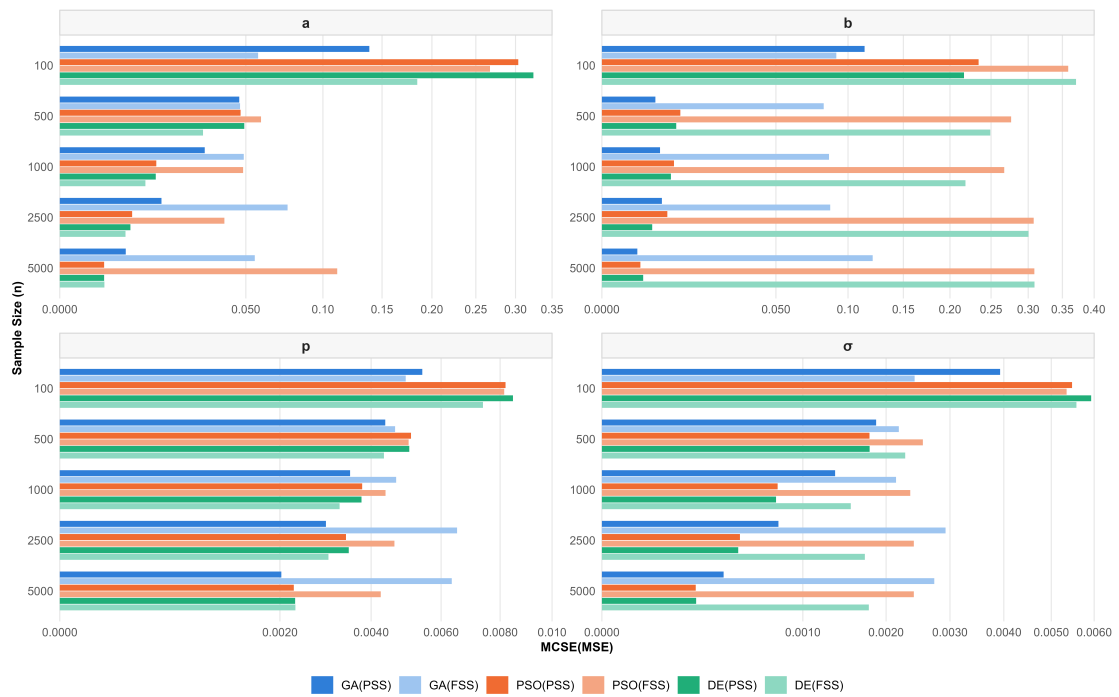


(c) Case III

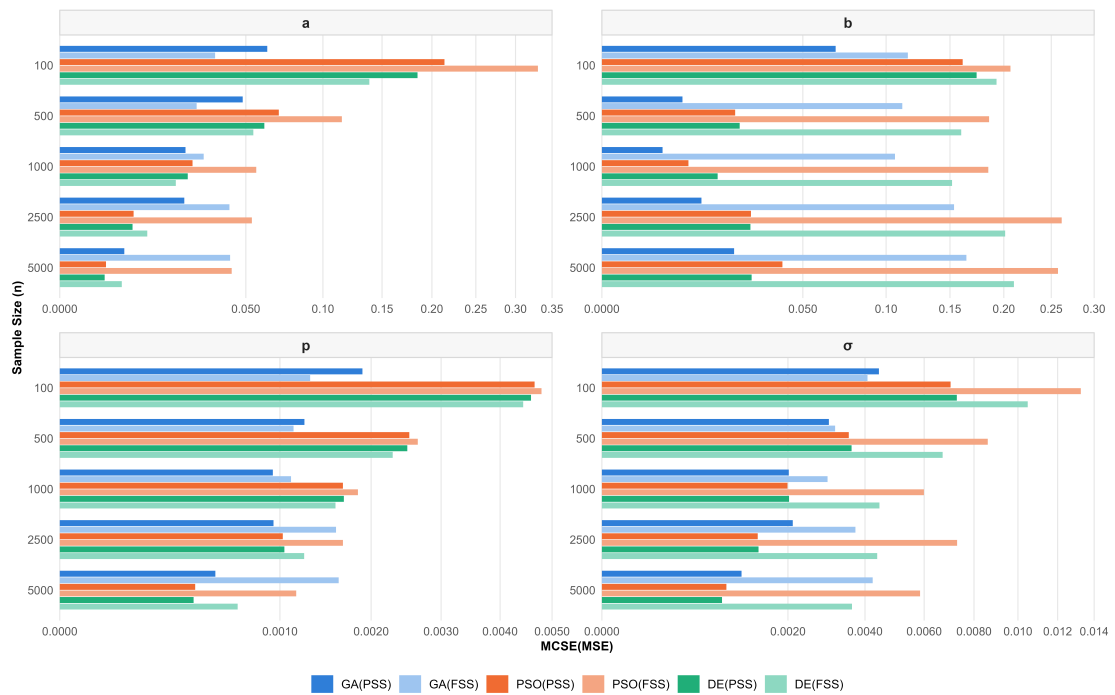


(d) Case IV

Figure A1. Monte Carlo standard errors of Bias values for the parameters a , b , p , and σ under Cases I–IV.



(a) Case I



(b) Case II

Continued on next page

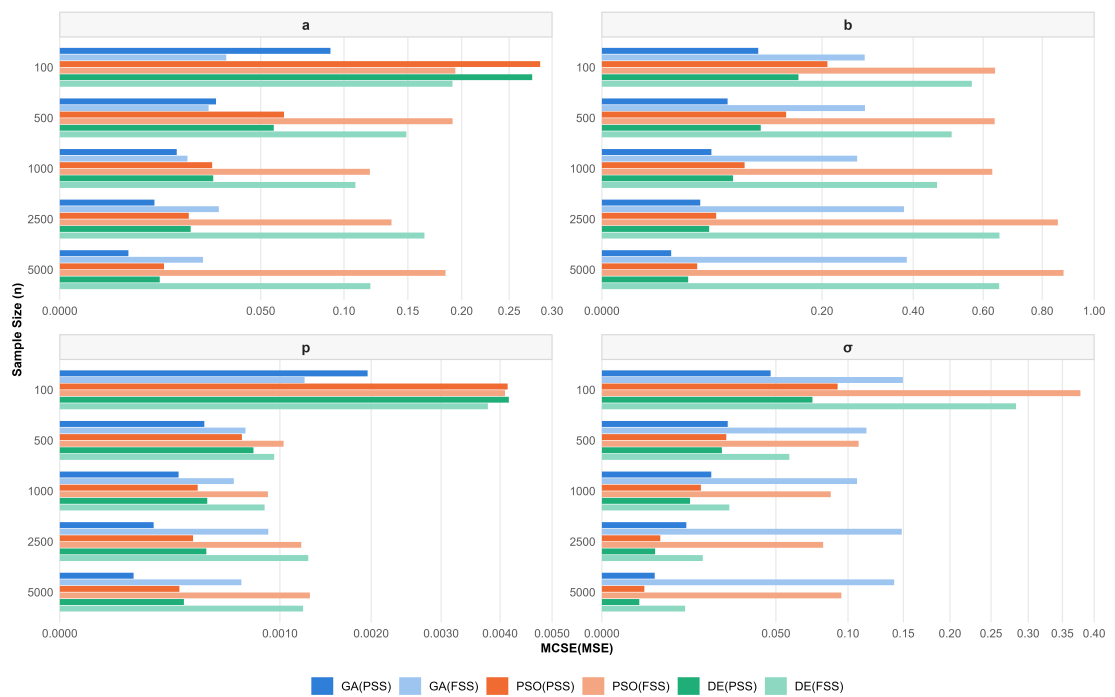
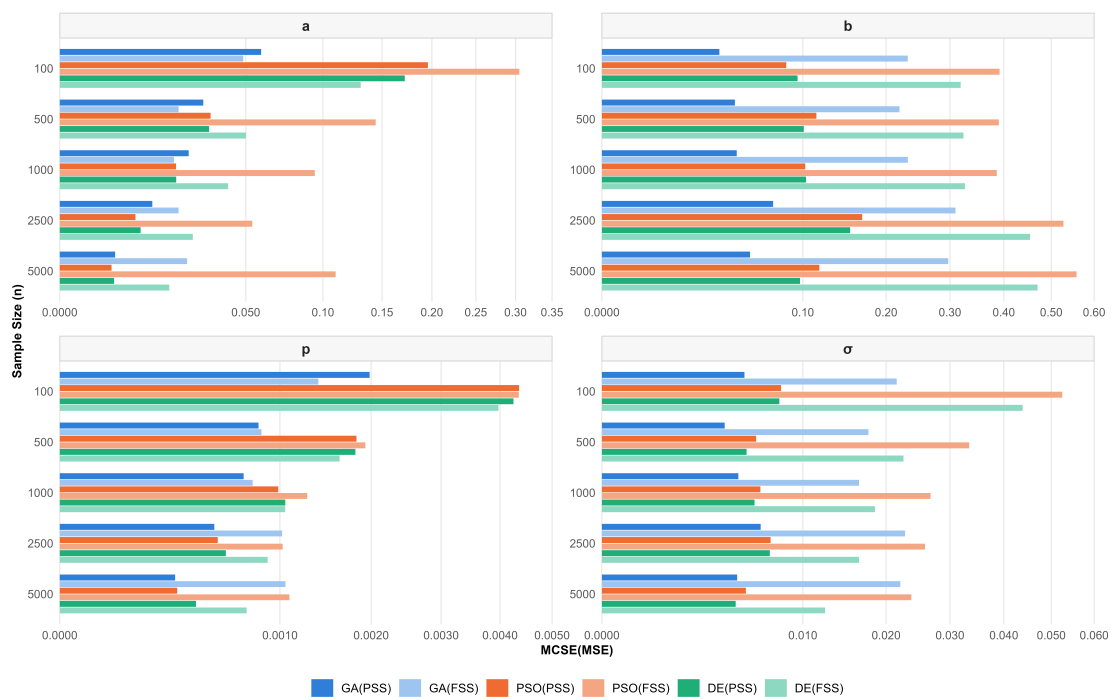


Figure A2. Monte Carlo standard errors of MSE values for the parameters a , b , p , and σ under Cases I–IV.

A.3. Computation time and convergence iteration results

Table A2. Mean computation time and average number of iterations required for convergence of the PSS and FSS-based GA, PSO, and DE algorithms across Cases I–IV.

Sample size (<i>n</i>)	Method	Case I		Case II		Case III		Case IV	
		Time (s)	Iter.	Time (s)	Iter.	Time (s)	Iter.	Time (s)	Iter.
100	GA(PSS)	17.81	83.50	17.56	86.22	19.30	80.35	16.05	82.80
	PSO(PSS)	17.89	74.65	17.65	77.70	19.39	78.62	16.16	83.70
	DE(PSS)	17.84	89.47	17.57	88.82	19.29	89.09	16.03	86.28
	GA(FSS)	0.58	85.53	0.57	79.99	0.62	78.76	0.58	80.71
	PSO(FSS)	0.68	84.96	0.66	86.01	0.73	84.43	0.70	84.88
	DE(FSS)	0.60	85.96	0.58	86.60	0.62	88.21	0.57	83.86
500	GA(PSS)	74.54	83.58	77.05	83.65	75.52	84.78	77.01	80.07
	PSO(PSS)	74.97	76.89	77.45	83.40	75.99	83.48	77.46	79.91
	DE(PSS)	74.89	86.73	77.40	85.55	75.90	88.15	77.33	89.99
	GA(FSS)	2.38	84.45	2.25	82.54	2.36	75.34	2.37	82.32
	PSO(FSS)	2.77	85.91	2.68	88.63	2.74	87.02	2.77	82.98
	DE(FSS)	2.70	84.97	2.63	87.19	2.74	84.80	2.72	83.05
1000	GA(PSS)	139.66	79.78	136.16	84.05	133.10	76.61	149.98	79.09
	PSO(PSS)	140.50	78.88	137.06	82.86	134.02	79.37	150.84	75.81
	DE(PSS)	140.45	86.81	136.90	88.15	133.82	89.03	150.81	86.68
	GA(FSS)	4.64	83.29	4.48	80.38	4.60	79.17	4.63	77.40
	PSO(FSS)	5.40	85.39	5.29	89.74	5.37	87.80	5.37	87.87
	DE(FSS)	5.47	87.21	5.27	87.30	5.32	84.83	5.38	81.78
2500	GA(PSS)	313.13	80.91	317.41	80.95	310.24	76.22	363.76	73.71
	PSO(PSS)	315.12	79.42	319.29	84.15	312.44	75.03	366.05	72.40
	DE(PSS)	315.05	81.60	319.25	88.08	312.36	87.41	365.90	87.64
	GA(FSS)	11.23	82.82	11.46	80.08	11.35	75.96	11.26	80.83
	PSO(FSS)	12.91	88.34	13.39	89.14	13.39	86.09	13.43	85.49
	DE(FSS)	13.24	85.23	13.38	84.79	13.60	84.92	13.47	81.47
5000	GA(PSS)	610.97	70.73	616.85	83.43	601.19	74.86	705.63	66.16
	PSO(PSS)	615.55	76.09	621.50	81.05	605.97	74.51	710.15	70.48
	DE(PSS)	615.39	82.54	620.80	88.49	605.93	86.76	709.73	82.09
	GA(FSS)	22.65	80.40	22.88	80.65	23.06	76.90	22.84	81.83
	PSO(FSS)	26.13	86.15	26.78	89.14	27.17	88.96	26.20	85.22
	DE(FSS)	26.97	87.59	27.33	87.86	27.22	85.15	27.13	81.28

Note: Time denotes the mean computation time in seconds, and Iter. denotes the average number of iterations required for convergence. For the PSS, computation time includes PSS construction and algorithm execution; for the FSS, it equals algorithm execution time.



AIMS Press

© 2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0>)