Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal, 2025, 19(12), 420–435
https://doi.org/10.12913/22998624/209887
ISSN 2299-8624, License CC-BY 4.0

Received: 2025.06.11
Accepted: 2025.10.01
Published: 2025.11.01

The usability of Bayesian filters in noise reduction across interdisciplinary data

Tomasz Zientarski[1*], Joanna Kozar[1], Kamil Pietrak[1]

[1] Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland
* Corresponding author's e-mail: t.zientarski@pollub.pl

ABSTRACT

Bayesian filters are most often used to predict the behaviour of dynamic objects in the presence of noise with a non-Gaussian distribution. Another application can be the filtration of measurement data obtained from measurement systems. Noise is present in almost all experimental data, and its distribution is often non-Gaussian. The article presents the application of Bayesian filtering methods to noisy data. For testing, real experimental data and artificially generated and noisy data with a known distribution were used. The following were used for testing: generic particle filter, SIR particle filter, auxiliary particle filter, and regularized particle filter. The effect of the number of inserted particles on the estimated result data was examined. The peak signal-to-noise ratio (PSNR) measure was used to assess the quality of the estimation. The results showed a significant advantage of the auxiliary particle filter over the other Bayesian filters. The same data sets were then subjected to Kalman filters. A basic and an extended Kalman filter were used. It turned out that all the Bayesian filters used, even for a small number of particles, give higher PSNR values than the commonly used extended Kalman filter.

Keywords: Bayesian filtering, particle filter, nonlinear filtering, Kalman filter, Monte Carlo methods

INTRODUCTION

In today's world, advances in technology and the increasing complexity of systems have made state estimation and signal filtering special issues in many scientific fields. Faced with the challenge of determining the precise state of systems under conditions of uncertainty and disturbance, estimation and filtering methods such as Bayes filters and Kalman filters have been developed. Bayesian filters, based on Bayes' theorem, make it possible to update probabilities based on the incoming data. They are widely used in localization [1-2], navigation [3-4], robotics [5], medicine [6], economics [7], Bayesian classifiers [8-9], and many others [10-11].

Paper [1] describes a method for dynamic estimation of the path attenuation exponent as a function of distance, based on actual received signal strength indicator (RSSI) measurements. The main goal was to develop a method for accurate distance estimation using a small measurement data set. For this purpose, a particle filter was used, allowing precise radio signal attenuation modeling. The sensitivity of the method to changing the number of particles and computational iterations was analyzed. The experimental results confirm the effectiveness of the proposed method in terms of improving the accuracy of path attenuation estimation. The method shows high utility in systems with limited hardware resources, making it attractive for sensor network (WSNs) applications and other wireless systems.

Paper [2] focuses on the problem of robot localization. Bayes filters were used to improve the results measured with the sensor, reducing the noise so that the results were closer to the real ones. This paper introduces an innovative method for the sensor model, referred to as the predictive sensor model, which incorporates a prediction mechanism to enhance the effectiveness of measurement updates in Bayesian filters. By adding sensor prediction, the original Bayes filter was extended to an anticipatory Bayes filter.

In-building navigation systems based on RSSI data are described in the article [3], in which the authors focus on the use of particle filters and Kalman filters based on Bayesian theory for state updating and improving, smoothing the measurement results. The paper mentions that particle filters and Kalman filters can be used in any case where the algorithm relies on updating the state of the particles.

Article [4] presents an application of Bayes' theory to data smoothing in a multi-target tracking application. The study uses a finite labeled dataset. It is shown that if the multi-target transition kernel used in the backward smoothing step does not take into account the birth and decay processes of the targets, the resulting smoothing density is identical to the filtering density. Simulations conducted by the authors in MTT scenarios indicate that smoothing performs better than a generalized labeled multi-Bernoulli filter in the context of an optimal subpattern assignment matrix.

Particle filters based on Bayes' theory are also used in robotics. Study [5], the author states that Bayes and Kalman particle filters are a standard approach in mobile robotics. The authors mainly focus on the robot's location and navigation using LIDAR laser sensors. For the experiment, they used a mobile robot with a laser sensor with a range of 0-25 m and an aperture angle of 270°. The authors compare particle filters with neural networks. The study results show that particle filters perform worse than trained neural networks, however, the latter require supervised learning and time for the training process itself, while filters can work immediately.

Another discipline where particle filters are applied is in medicine. This is described in the article [6], where the authors focus on the ECG signal. They note that the ECG signal used in noninvasive cardiac electrophysiology is a very convenient and helpful tool. Unfortunately, most often the signal is noisy. They emphasize that data cleanliness is crucial in the further study of the obtained signal. They use Bayesian methods to average the weighted received signal from the measuring device. Ultimately, the study shows that the use of particle filters significantly improves, reducing noise in the ECG signal.

The next paper related to economics is [7]. It describes the use of Bayesian filtering in electricity forecasting, with a focus on prioritization to improve forecasting accuracy. The authors review the general structure of Bayesian forecasting, highlighting the computational techniques used to implement the approach.

The authors [8] focused on the use of the naive Bayes (NB) classifier, which was applied to data from social networks. In order to analyze these data, a network version of the naive Bayes classifier, which is an extension of the classical NB model, was proposed. The statistical properties of the NNB model were theoretically analyzed, and its effectiveness was evaluated through simulations. In addition, an analysis of real data was carried out, which confirmed the practical application of the method.

In [9], the authors focused on applying a naive Bayes classifier to predict the risk of contracting and developing type 2 diabetes. The analysis was carried out on the Pima Indians Diabetes Data Set, which contains information on people with type 2 diabetes, as well as healthy individuals. The results show that the model used was highly effective, confirming that the Bayes network can effectively predict type 2 diabetes.

The authors of this paper [10] focus on email spam detection. They propose using a hybrid spam detection technique combining Naive Bayes and Markov Random Field algorithms. Naive Bayes identifies spam using probabilistic classification based on Bayes' theorem, while Markov Random Field models the statistical dependencies of spam patterns. This approach improves spam detection performance in terms of accuracy and processing time.

The paper discusses various approaches to learning and implementing Bayesian Network (BN) classifiers [11]. The authors evaluated algorithms of four different types of BN classifiers and determined their effectiveness compared to established methods. Thie study highlights that they offer a powerful alternative to other types while maintaining reasonable computational requirements.

In paper [12], a method for improving the quality of biomedical ECG signals based on nonlinear dynamic models and Bayesian filtering was used. The proposed approach is based on a realistic, synthetic ECG model, which was used in the extended Kalman filter (EKF) and its smoothed variant (EKS) and the unscented Kalman filter (UKF). The applied solution gave better results than classical filtration techniques in a wide range of SNR values. This approach allows not only to preserve the signal morphology, but also to effectively track its changes in low SNR conditions.

In [13], a PPG signal filtering using a particle filter was proposed to improve the accuracy of emotion recognition in wearable systems. The applied approach enables effective signal tracking in the presence of typical dynamic typical motion disturbances. The conducted experiments showed an increase in emotion recognition accuracy by an average of 11.8% and an improvement in SNR by 4.5 dB compared to the baseline NLMS method.

In [14], a comprehensive comparison of two Bayesian image denoising methods Total Variation (TV) and Bayesian least squares – Gaussian scale mixture (BLS-GSM) was presented. The authors showed that the TV-based approach better preserves sharp edges and fine anatomical structures, while the wavelet method (BLS-GSM) provides higher global image quality, especially in low-frequency regions. The studies were conducted on both simulated images and real dental radiographs. The authors' conclusions suggest that there is no universal solution, the choice of the algorithm should be adapted to the image characteristics and the expected compromise between noise reduction and detail preservation.

The main purpose of the present work is to perform a detailed analysis of the suitability of various Bayesian filters for use with different types of experimental data. Four commonly used Bayesian filters in standard versions were chosen for testing: generic particle filter, SIR particle filter, auxiliary particle filter, and regularized particle filter. The peak signal-to-noise ratio measure was used to assess the quality of the level of noise reduction. Next, the results obtained were compared with those obtained using Kalman filters.

BAYESIAN FILTERING METHODS

Particle filters are based on Thomas Bayes' theorem, which is a fundamental theorem in probability theory. The theory makes it possible to calculate conditional probabilities, hence allowing updating the probabilities with the updated data. The theorem is presented in formula (1)

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{1}$$

where: $P(A|B)$ – probability of the occurrence of event A under the condition that event B occurred (conditional probability), $P(B|A)$ – probability of event B under the condition that event A has occurred, $P(A)$ – a priori probability of event A, before considering event B, $P(B)$ – probability of event B.

In the case of the filters discussed in the article, this is the basis for tracking the state (x_k) of dynamic systems when the system is nonlinear or the noise does not come from a Gaussian distribution. Bayes' theorem makes it possible to calculate the posterior probability, which is shown in the formula (2)

$$P(x_k|z_{1:k}) = \\ = \frac{P(z_k|x_k) \cdot P(x_k|z_{1:k-1})}{P(z_k)} \tag{2}$$

where: z denotes measured data, x – hidden state of system, $P(x_k|z_{1:k})$ – a posteriori distribution of the state, $P(z_k|x_k)$ – reliability function, $P(x_k|z_{1:k-1})$ – a priori distribution of the state, and $P(z_k)$ is a normalizer that ensures that the result is a probability.

Particle filters consist of three phases:

- prediction: updating the state of each particle based on the system dynamics model,
- updating: changing the particle weights based on new observations and Bayes' theorem,
- resampling: removing particles with high weights to better represent the posterior distribution.

Resampling algorithm

The systematic resampling algorithm [15] shown in Algorithm 1 was used in all particle filters. This is one of the possible implementations of the algorithm for performing resampling. This implementation was chosen because it is the most widely used in other research works [1, 16-17]. The task of the resampling process is to remove particles that represent low weight and focus on those with high weight.

During the resampling process, new particles are created by calculating them from an approximate discrete representation defined by:

$$p(x_k \mid z_{1-k}) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i) \tag{3}$$

Used in the algorithm, the variable CDF for the following particles is calculated from:

Algorithm 1: Resampling algorithm

input : x_k-particles values array for k point
w_k-particles weight array for k point
ind-indices array
N-number of particles,

Create a cumulative distribution function array CDF
Set $CDF_0 = 0$
for $k = 1$ **to** $N - 1$ **do**
  Assign rest CDF using (4)
end
Draw random starting point $u_1 \sim \mathbb{U}\,[0, N_s^{-1}]$
Initialize $i = 1$
for $j = 0$ **to** $N - 1$ **do**
  Compute u using (5) .
  while $u > CDF^i$ & $i < N - 1$ **do**
    Increment i
  end
  Assign $ind^j = i$
end
for $j = 0$ **to** $N - 1$ **do**
  Resample and reasign particles
  $x_k^j = x^{ind^j}$
  $w_k^j = 1/N$
end

$$\mathrm{CDF_k} = \mathrm{CDF_{k-1}} + \mathrm{w_k} \tag{4}$$

The average (u) for each particle is calculated from:

$$\mathrm{u} = \mathrm{u_1} + \mathrm{j/N} \tag{5}$$

Sequential importance sampling

Sequential importance sampling (SIS) shown in Algorithm 2 is a basic sequential Monte Carlo sampling method that allows efficient sampling from complex probability distributions by sequentially updating sample weights over time using Equation (6):

$$\mathrm{w_k^i} \propto \mathrm{w_{k-1}^i}\,\frac{\mathrm{p}\left(\mathrm{z_k} \mid \mathrm{x_k^i}\right)\mathrm{p}\left(\mathrm{x_k^i} \mid \mathrm{x_{k-1}^i}\right)}{\mathrm{q}\left(\mathrm{x_k^i} \mid \mathrm{x_{k-1}^i}, \mathrm{z_k}\right)} \tag{6}$$

The SIS aims to overcome the limitations of traditional Monte Carlo methods for complex engineering models and rare failure events. SIS works by generating samples that progressively approach the optimal sampling density. This sequential approach helps efficiently sample the failure domain. Article [18] emphasizes how SIS can improve the proposal distribution by using information from previous iterations, making

Algorithm 2: Sequential Importance Sampling

input : z - One dimensional measured data array
N_s - number of experimental points,

output: x - calculated values array
w - assigned weight array

for $i = 0$ **to** N_s **do**
  Draw $\mathbf{x}_k^i \sim q\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^i, \mathbf{z}_k\right)$
  Assign the particle weight w_k^i using equation (6)
end

it well-suited for structural reliability problems where the failure is complex or difficult to characterize in advance.

Generic particle filter

The generic particle filter (GPF) presented in Algorithm 3 is based on the sequential importance sampling filter. SIS is the base filter in terms of construction; however, it has several problems solved in GPF. The first problem is degeneration, which means that a negative value can be assigned to the importance weight after several iterations. The paper [19] shows that the variance of importance weights can only increase, thus eliminating the degeneration problem. Another problem is the selection of the correct importance weight. In this case, some methods can be used, described in detail in paper [16]. One of them was chosen for the study, where importance density is a priori. Finally, resampling is the creation of new samples based on calculated weights. For the purposes of this study, systematic resampling was used in [15] as presented in Algorithm 1. This resulted in a filter based on SIS, but without its drawbacks.

The initial value of the particles is assigned using equation (7)

$$x_k \sim \mathcal{N}\ (init\_state,\ init\_variance) \tag{7}$$

Initial weights are calculated using equation (8)

$$w_k = 1/N \tag{8}$$

Next weights can be calculated using equation (6), and to ensure that the problem of degeneration does not affect the results, resampling was used only if N_{eff} shown in formula (9) goes above some threshold N_T

$$N_{eff} = \frac{1}{\sum_{j=1}^{N} w_k^j} \tag{9}$$

Finally, resampled values are calculated using formula (10)

$$x_k = \sum_{j=0}^{N} x_k^j \cdot w_k^j \tag{10}$$

Correct selection of the N_T threshold [20-22] is another crucial task to balance computational cost and filter performance. The most common methods for selecting the value of N_T are the relative threshold, the fixed threshold, and the adaptive threshold. The value of N_T also determines

Algorithm 3: Generic Particle Filter

input : z - One dimensional measured data array
N_s - number of experimental points,
N - number of added particles,
N_T - threshold for resampling steps,

output: x - calculated values array

for $k = 0$ **to** $N - 1$ **do**
  Prepare N random particles and assign base values (7) and base weights (8)
end
for $k = 0$ **to** $N_s - 1$ **do**
  for $i = 0$ **to** $N - 1$ **do**
    Draw $\mathbf{x}_k^i \sim q\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^i, \mathbf{z}_k\right)$
    Assign the particle weight using equation (6)
  end
  Calculate total weight: $w_s = \sum_{i=1}^{N} w_k^i$
  for $i = 0$ **to** $N - 1$ **do**
    Normalize weights: $w_k^i = w_k^i / w_s$
  end
  Calculate N_{eff} using (9)
  if $N_{eff} > N_T$ **then**
    Resample particles using algorithm (1)
  end
  Assign denoised value x_k using weighted mean (10)
end

the problem of particle degeneration. Too low a threshold value will cause degeneration to occur, but it will reduce the execution times of the algorithm. A large threshold value will zap many samples by triggering resampling more frequently, increasing the computational cost.

The relative threshold method is to select the threshold as a fraction of the selected number of particles. A popular choice is $N_T = \alpha N$, where $\alpha = [0.5, 0.8]$. This fraction says that resampling is triggered when αN particles make an effective contribution.

The fixed threshold method involves determining N_T values based on empirical studies or specific application requirements. This approach provides consistent behavior across runs, making it easier to compare performance metrics. However, it may not adapt well to dynamic changes in the system or varying levels of noise.

The adaptive threshold method involves selecting N_T values based on application noise or system state in real time. It dynamically adjusts N_T during runtime, which helps maintain a balance between accuracy and efficiency under changing conditions. This method often relies on statistical measures such as variance or entropy to determine the optimal threshold of N_T choice.

SIR particle filter

SIR particle filter (sampling importance resampling), which was presented in papers [16, 23], can be applied to recursive Bayesian filtering using the Monte Carlo (MC) method. The SIR Particle Filter is a modification of the SIS algorithm, in which the method of counting the density of importance is chosen from the available options. In this case, the density of priority is calculated using the prior (11)

$$q\left(x_k \mid x_{k-1}^i, z_k\right) = \\ = p\left(x_k \mid x_{k-1}^i\right) \tag{11}$$

Then substitution of (11) into (6) gives equation:

$$w_k^i \propto w_{k-1}^i p\left(z_k \mid x_k^i\right) \tag{12}$$

However, given that resampling is performed at each time index where $w_{k-1}^i = 1/N \forall i$, this formula can be simplified to formula (13)

$$w_k^i \propto p\left(z_k \mid x_k^i\right) \tag{13}$$

Another change compared to SIS is that resampling is done at each time index.

Algorithm 4: SIR Particle filter

input : z - One dimensional measured data array
N_s-number of experimental points,
N-number of added particles,

output: x - calculated values array

for $k = 0$ **to** $N - 1$ **do**
  Prepare N random particles and assign base values (7) and base weights (8)
end
for $k = 0$ **to** $N_s - 1$ **do**
  for $i = 0$ **to** $N - 1$ **do**
    Draw $\mathbf{x}_k^i \sim p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^i\right)$
    Calculate $w_k^i = p\left(\mathbf{z}_k \mid \mathbf{x}_k^i\right)$ (13)
  end
  Calculate total weight: $w_s = \sum_{i=1}^{N} w_k^i$
  for $i = 0$ **to** $N - 1$ **do**
    Normalize weights: $w_k^i = w_k^i / w_s$
  end
  Resample particles using algorithm (1)
  Assign denoised value x_k using weighted mean (10)
end

Auxiliary particle filter

Auxiliary particle filter (APF) is another iteration of the SIR filter presented in the paper [24]. This filter can be derived from the baseline SIS by introducing the importance density $q(\mathbf{x}_k, i|\mathbf{z}_{1:k})$, which operates on $\{x_k^j, i^j\}_{j=1}^{M_s}$ where i refers to the index in $k-1$ and M_s refers to number of samples generated as a result of resampling. After using Bayes' theorem, and introducing the mean (μ_k^i) shown in equation (14)

$$\mu_k^i = E\left[x_k \mid x_{k-1}^i\right] \tag{14}$$

where: E denotes an expected value operator, x_k – state in stem k, x_{k-1}^i – state of *i-th* particle at previous step.

The following equation, shown in (15), was reached:

$$\begin{aligned} q(x_k, i \mid z_{1:k}) = \\ = q(i \mid z_{1:k}) q(x_k \mid i, z_{1:k}) \end{aligned} \tag{15}$$

which, after the necessary transformations described in [16], finally looks as shown in equation (16):

$$\begin{aligned} w_k^j \propto w_{k-1}^{ij} \frac{p\left(z_k \middle| x_k^j\right) p\left(x_k^j \middle| x_{k-1}^{ij}\right)}{q\left(x_k^j, i^j \middle| z_{1:k}\right)} = \\ = \frac{p(z_k \mid x_k^j)}{p(z_k \mid \mu_k^{ij})} \end{aligned} \tag{16}$$

After transformations, the algorithm looks as shown in Algorithm 5.

Algorithm 5: Auxiliary Particle filter

input : z - One dimensional measured data array
N_s-number of experimental points,
N-number of added particles,

output: x - calculated values array

for $k = 0$ **to** $N - 1$ **do**
  Prepare N random particles and assign base values (7) and base weights (8)
end
for $k = 0$ **to** $N_s - 1$ **do**
  for $i = 0$ **to** $N - 1$ **do**
    Calculate μ_k^i using formula (14)
    Calculate weights $w_k^i = q(i \mid \mathbf{z}_{1:k}) \propto p(\mathbf{z}_k \mid \mu_k^i)\, w_{k-1}^i$
  end
  Calculate total weight: $w_s = \sum_{i=1}^{N} w_k^i$
  for $i = 0$ **to** $N - 1$ **do**
    Normalize weights: $w_k^i = w_k^i / w_s$
  end
  Resample particles using algorithm (1)
  for $j = 0$ **to** $N - 1$ **do**
    Draw $\mathbf{x}_k^j \sim q(\mathbf{x}_k \mid i^j, \mathbf{z}_{1:k}) = p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{ij}\right)$ as in SIR Particle Filter
    Assign weight w_k^j using algorithm (16)
  end
  Calculate total weight: $w_s = \sum_{i=1}^{N} w_k^i$
  for $i = 0$ **to** $N - 1$ **do**
    Normalize weights: $w_k^i = w_k^i / w_s$
  end
  Assign denoised value x_k using weighted mean (10)
end

Regularized particle filter

The regularized particle filter (RPF) [25-26] is an enhanced version of the generic particle filter, where the resampling process has been refined and expanded. The modification includes the incorporation of variance calculation and the application of the Epanechnikov Kernel. The variances (S_k) are computed using equation (17). A detailed presentation of the RPF can be found in Algorithm 6

$$S_k = Var\left(\{x_k^i, w_k^i\}_{i=1}^{N_S}\right) \tag{17}$$

and then standard deviation in step (D_k) was derived using equation (18) because the experiment was conducted for one-dimensional data

$$D_k = \sqrt{S_k} \tag{18}$$

Then the previously used resampling algorithm was used, after which a value was generated for each sample using the Epanechnikov Kernel (K(∈)), described by the equation (19) for one-dimensional data

$$K(\epsilon) = \frac{3}{4}(1-\epsilon^2), \quad |\epsilon| \leq 1 \tag{19}$$

Finally, the value of the particle is updated according to the equation:

$$x_k^i = x_k^i + h_{opt}\sigma_w^2\epsilon^i \tag{20}$$

where: h_{opt} denotes optimal bandwidth used for smoothing the particle distribution, σ_w^2 – representance of variance of the particle weights, $\in^i$ – sample drawn from the Epanechnikov Kernel function described by equation (19)

Algorithm 6: Regularized Particle filter

input : z - One dimensional measured data array
N_s-number of experimental points,
N-number of added particles,
N_T-treshold for resampling steps,

output: x - calculated values array

for $k = 0$ **to** $N-1$ **do**
  Prepare N random particles and assign base values (7) and base weights (8)
end
for $k = 0$ **to** $N_s - 1$ **do**
  for $i = 0$ **to** $N-1$ **do**
    Draw $\mathbf{x}_k^i \sim q\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^i, \mathbf{z}_k\right)$
    Assign the particle weight using equation (12)
  end
  Calculate total weight: $w_s = \sum_{i=1}^{N} w_k^i$
  for $i = 0$ **to** $N-1$ **do**
    Normalize weights: $w_k = w_k / w_s$
  end
  Calculate N_{eff} using (9)
  if $N_{eff} > N_T$ **then**
    Calculate variance σ_w^2 of $\{\mathbf{x}_k^i, \quad w_k^i\}_{i=0}^{N-1}$ (17)
    Compute D_k (18)
    Resample particles using algorithm (1)
    for $i = 0$ **to** $N-1$ **do**
      Draw $\epsilon^i \sim K$ from the Epanechnikov Kernel (19)
      Assign $\mathbf{x}_k^i = \mathbf{x}_k^i + h_{opt}\sigma_w^2\epsilon^i$
    end
  end
  Assign denoised value x_k using weighted mean (10)
end

EXPERIMENT

For this work, four versions of programs were written in C to implement the four filters described before. The programs were compiled using standard compiler settings. The free and open-source C/C^{++} GNU Compiler in version 13.2.0 was used. The computer used for the experiment is built with the following components: an Intel Core i7 8700K processor, 32GB DDR4 memory, and an SSD disk.

The four test data sets were selected. These datasets contain real measurement data from various experiments, comprising several dozen to several thousand measurement points. Some of them came from IoT devices. It was decided to reduce the data sets to 60 measurement points (N_s=60) to enable their mutual comparison for testing. Moreover, from the original data sets, fragments were selected that reflect the typical behavior of the actual measurements. For later use, the data sets were named ECG, EEG, RSSI, STRESS, TTL and XRD for electrocardiographic data, electroencephalographic data, received signal strength indicator data, mean biaxial stress data, oscilloscope data and X-ray diffraction data, respectively.

- The electrocardiographic data describe the heart's performance using electrical signals. The data are taken from the following source: https://www.kaggle.com/datasets/protobioengineering/mit-biharrhythmia-database-modern-2023.
- The electroencephalographic data describe the electrical activity of the brain. The datasets were recorded for C3-A1 and C4-A1 electrodes. The data are taken from https://www.kaggle.com/datasets/jbouv27/eeg.
- The Received Signal Strength Indicator measures the power present in a received radio signal. In this case, the RSSI represents the real measurements of radio signal data between two ZigBee nodes. The measurements were carried out for a distance of 15 meters. The data are taken from paper [27].
- The dataset represents simulation data of the evolution of the mean biaxial stresses during the deposition of Co atoms on the Cu surface. The deposition rate equals one atom 1per 2000 time steps, T = 300K, and the deposition energy is 1 eV. The data are taken from [28].
- The real oscilloscope data (TTL, transistor-transistor logic) represents a slice of the astable flip-flop waveform. These are values in the range of 0-5V. This is the standard operating voltage of transistors made with this technology. Authors' own data.
- The part of the X-ray diffraction profiles for the Fe-C-Mn-B-Cr-Si alloys quickly cooled in water. The data are taken from the paper [29].

RESULTS

In this work, we perform a detailed analysis of the suitability of various Bayesian filters for reconstructing measurement data, which is always subject to some noise and uncertainty. Six different experimental datasets were selected for analysis. These data represented different levels of waveform variability, which enabled the evaluation of how individual filters performed under different conditions. Both cases with relatively stable characteristics and those with significant fluctuations in measurement values over time were included. This approach allowed for a comprehensive evaluation of the effectiveness of selected filtration methods and their potential application in real conditions. The series of original experimental data are shown in Figure 1. Each data set contains 60 measurement points (N_s=60).

This dataset is used as the input file for filtering algorithms. The results, which contained the denoised input data, were saved during the calculations, and the execution times were measured. The choice of the number of inserted particles is critical for optimal results. Too many of them improve the estimation error but increase the calculation time. Too few of them speed up the estimation time, but give erroneous estimates. The calculations were performed for a number of inserted particles equal to 100, 500, 1000, and 2000.

The computational cost of calculation is directly proportional to the number of experimental points and the number of inserted particles and is $O(Ns \cdot N)$. Table 1 presents the average running times of selected filters. The data shows that the GPF and RPF algorithms are executed much faster than the others. This proves the correctness of the dependence of the particle resampling process on the value of the N_T parameter. Therefore, start-up times are not comparable.

Real data directly obtained from an experiment are always characterized by some level of noise. A commonly used statistical measure for determining the scatter of experimental data is

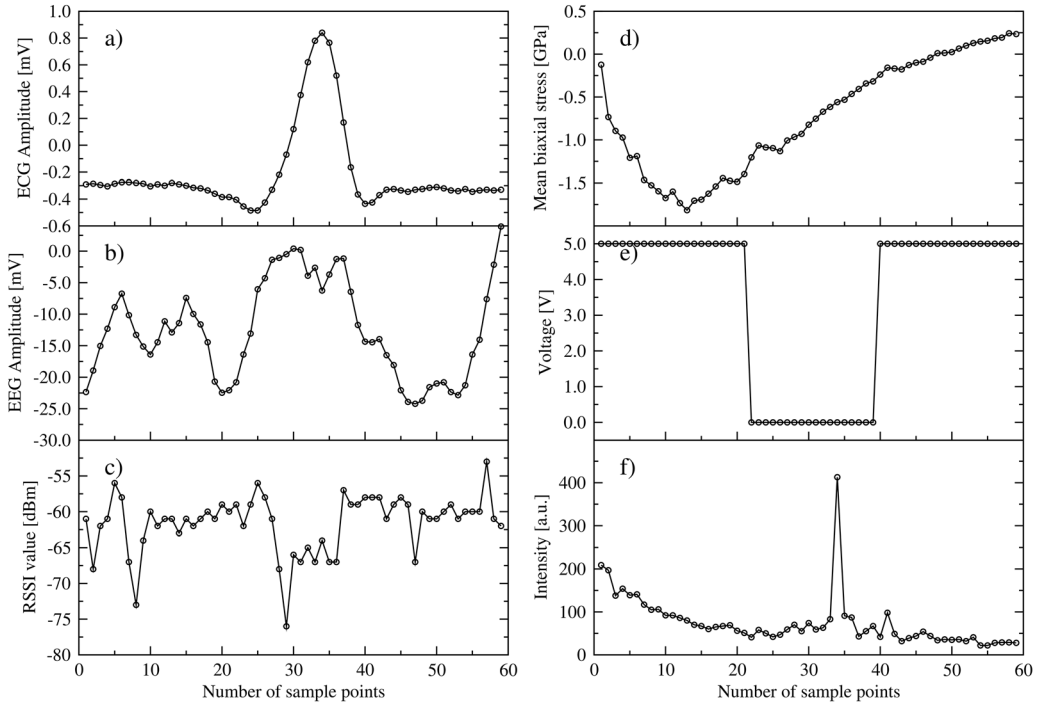

Figure 1. The curves in the graph represent the electrocardiographic data (a), electroencephalographic data (b), received signal strength indicator data (c), mean biaxial stress data (d), oscilloscope data (e), and X-ray diffraction data (f), respectively.

Table 1. The average execution time of selected filters for a number of inserted particles equals 100, 500, 1000, and 2000. Each data set contained only 60 measurement points

Filter type	Average execution time [ms] for a number of particles equal to			
	100	500	1000	2000
Generic particle filter	0.38	3.51	7.10	13.48
SIR particle filter	1.28	6.24	12.40	22.27
Auxiliary particle filter	1.20	5.50	10.79	19.59
Regularized particle filter	0.77	3.90	7.62	14.57

the standard deviation [30–32]. The standard deviation values for each data set are calculated and shown in Table 2. Moreover, the standard deviation values given in Table 2 were used as a process noise parameter (σ) in our Bayesian filters.

The standard deviation is a statistical measure that indicates the statistical spread of experimental data [33-35]. When looking at the data presented in Table 2, it is not possible to compare them directly. The standard deviations in the table appear in different units and have different values. Another important error assessment coefficient is the root mean square error [36-38]. This coefficient also does not allow for comparing numerically divergent results (different ranges of min and max values). With such divergent values of the analyzed numerical waveforms, a measure is needed that will give normalized values.

A commonly used measure to determine the quality of lossy compressed images is the Peak signal-to-noise ratio (PSNR) [39-45]. The PSNR is a measure that compares the maximum level of the expected signal to the level of the accompanying noise. Originally, PSNR is defined as:

$$PSNR = 10 \cdot log_{10} \frac{D^2}{MSE} \tag{21}$$

where: D^2 is the dynamical intensity range.

For an 8-bit image it is 256, and MSE is the mean squared error. In our case, we used a modification of the PSNR definition as follows:

Table 2. The standard deviation of the experimental data.

ECG [mV]	EEG [mV]	RSSI [dBm]	STRESS [GPa]	TTL [V]	XRD [a.u.]
0.311	8.016	4.071	0.660	2.291	59.445

$$PSNR = 10 \cdot \log_{10} \frac{(P_{max} - P_{min})^2}{MSE} \quad (22)$$

where: P_{min} and P_{max} determine the minimum and maximum values of the experimental data.

This definition of PSNR introduces normalization of the obtained values and allows comparison of the results obtained for all considered experimental data.

The selected Bayesian filters were implemented in C. The natural feature of the filters used in the scientific experiment is the use of pseudorandom numbers. Naturally, the built-in SRAND generator was used. The PSNR coefficient was calculated for each dataset. Figure 2 shows the obtained results only for the RSSI data.

Figure 2a shows the PSNR results obtained for different numbers of inserted particles for different types of filters. Initially, we observe an increase in the PSNR value with the increase in the number of particles for all types of filters. A low number of inserted particles does not guarantee a correct estimation of the RSSI value. Hence, it has a low PSNR value. Gradually, with the increase in the number of inserted particles, the PSNR value reaches a constant value. Unfortunately, for the APF, there are quite large fluctuations in the PSNR value. After analyzing the algorithm, it turned out that these fluctuations are caused by a poor quality of the generator implemented in C (SRAND) [46]. The APF and SPF filters use many more and more frequent pseudorandom numbers with respect to the other filters. Changing the generator to RAN2, as described in Numerical Recipes [47], significantly improved the results for all used filters. Figure 2b shows the same results obtained using a new pseudorandom number generator.

It is worth noting that changing the generator seed to another value causes a visible change in the PSNR value only for a low number of inserted

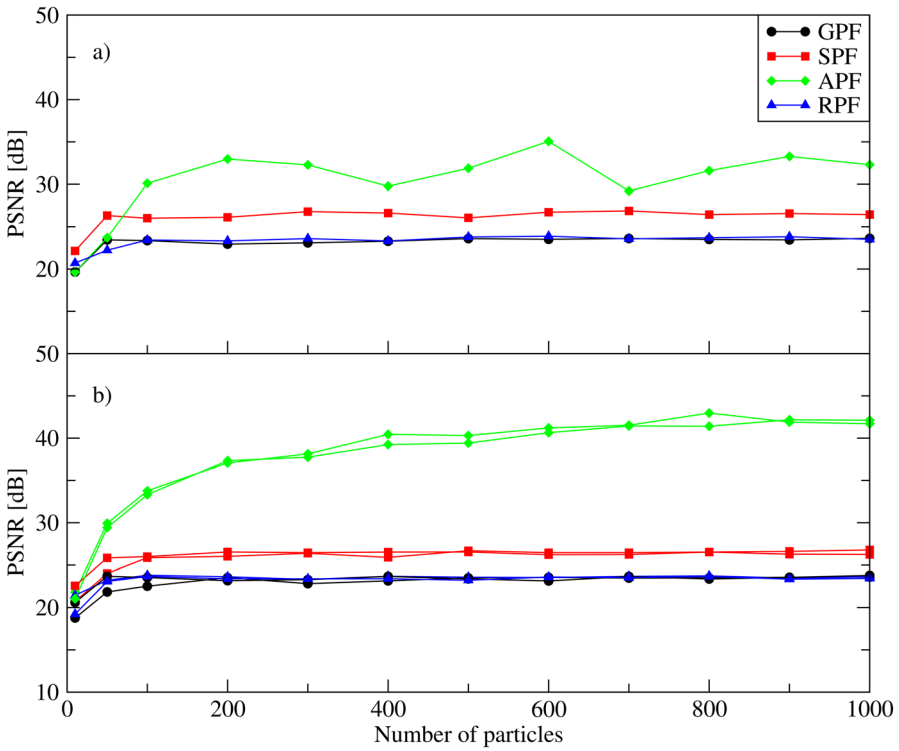

Figure 2. PSNR coefficient for RSSI data for different numbers of inserted particles and different types of filters. In part (a), the built-in pseudorandom number generator (SRAND) was used. In part (b), the RAN2 generator was used with two different seeds.

particles (below 200). Above this value, the largest PSNR fluctuations are observed for APF, and they amount to a maximum of 2 dB.

Finally, Figure 3 presents the PSNR results obtained for various numbers of inserted particles, filters, and experimental data types. In all observed cases, GPF, SPF, APF and RPF for small N values give low PSNR values. With the increase of the number of inserted particles, the PSNR value increases, reaching a steady state value.

Another noteworthy relationship is the significant increase in PSNR values obtained by the APF for EEG, RSSI, TTL and XRD data. In the case of ECG and STRESS data, the APF filter generates lower PSNR values similar to those obtained by the SPF filter. The APF filter achieves the highest filtering efficiency in the case of data characterized by rapid dynamics of signal value changes. The above observations are confirmed in Figure 4, which presents the original and estimated data obtained from the APF and SPF

The results reveal that the calculated values approximate the experimental data very well, increasing the PSNR value. The APF filter estimates data very well, especially at the points of their rapid change. This is well visible for TTL and XRD data, but also for others.

It is interesting from the point of view of the usefulness of the Bayesian filters used in this work to compare them with other types of filters. In our work, we compared them with the Kalman filter and the extended Kalman filter, using the classical configuration provided in paper [48]. It is common knowledge that the Kalman filter (KF) gives correct results for linear models. Unfortunately, most phenomena in the real world are characterized by nonlinearity. In such cases, the Kalman filter becomes useless. An alternative to the Kalman filter is the extended Kalman filter (EKF), which transforms the nonlinear problem into a linear one using the first-order Taylor approximation before applying the Kalman filter.

Figure 5 presents the PSNR values obtained from the analyzed data using particle and Kalman filters. We observe that a small number of particles (N=500) allows achieving higher PSNR values than Kalman filters. All particle filters used in this work provide better estimation of experimental data than even the extended Kalman filter. It is

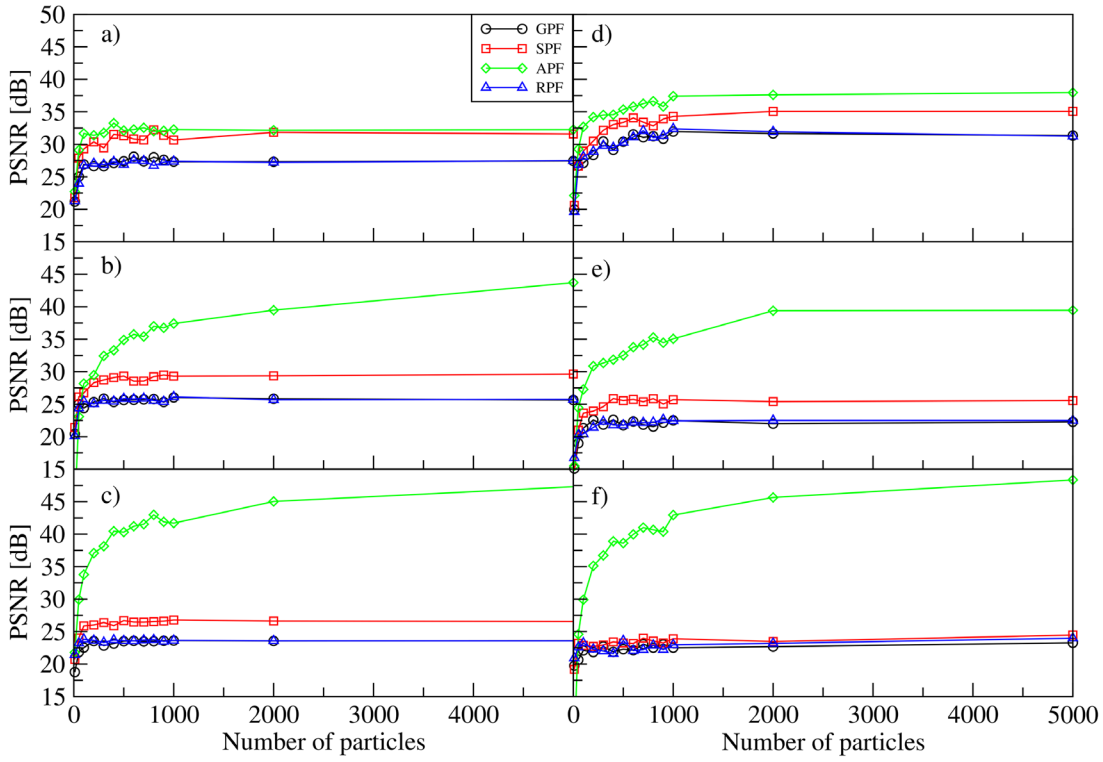

Figure 3. PSNR coefficient for different numbers of inserted particles and different data types: ECG (a), EEG (b), RSSI (c), STRESS (d), TTL (e) and XRD (f), respectively.
The types of filters are provided in the legend box. The RAN2 generator was used.

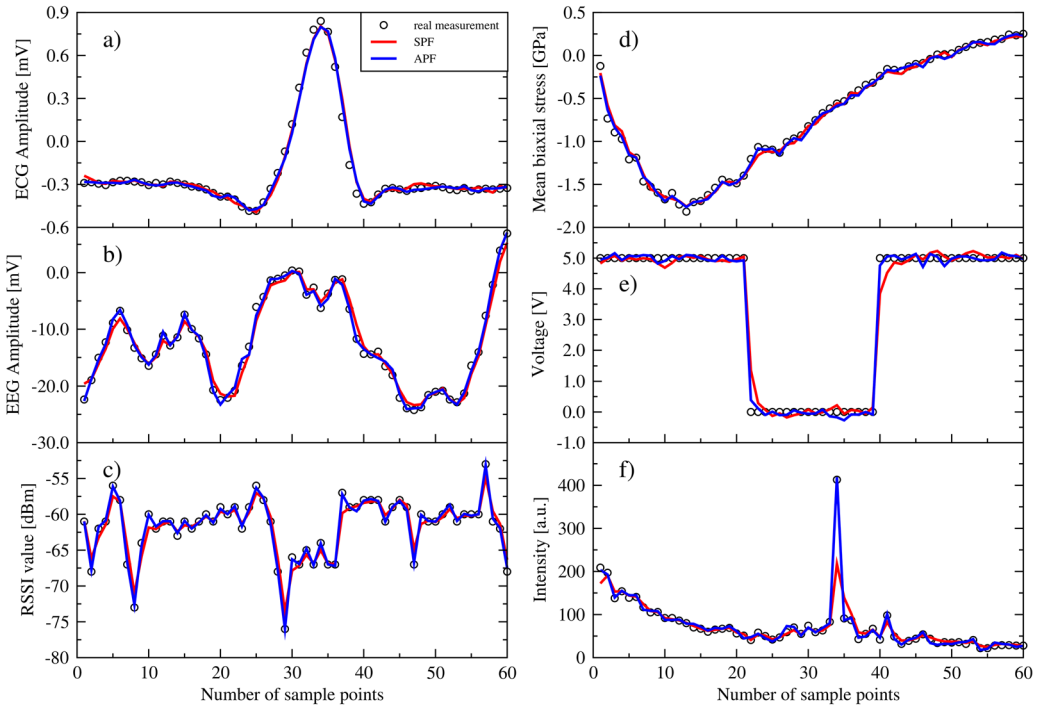

Figure 4. The real experimental data and estimated particle filter data for the number of inserted particles equal to 500. The APF and SPF filters are used. The curves in the graph represent the ECG data (a), EEG data (b), RSSI data (c), STRESS data (d), TTL data (e), and XRD data (f), respectively.

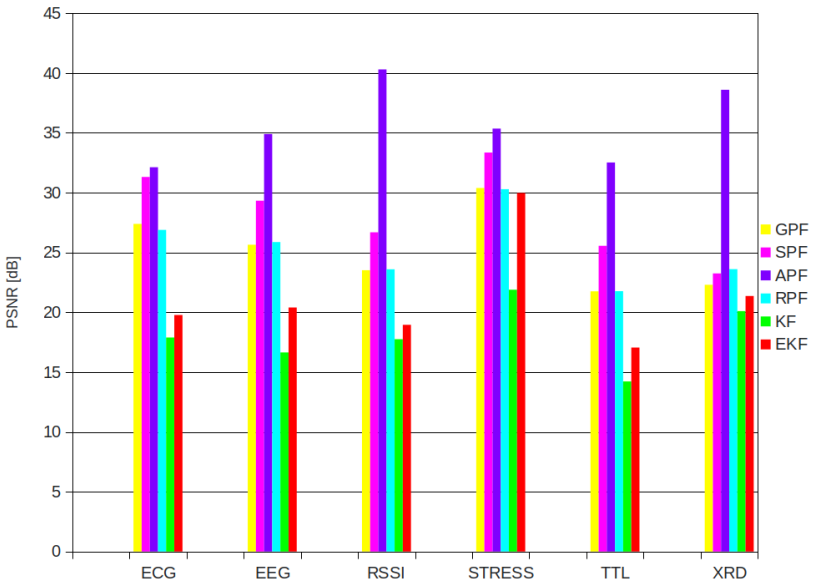

Figure 5. Comparison of PSNR coefficient values for different types of filters and different data types. The types of filters are provided in the legend box. For particle filters, the number of inserted particles was 500.

also worth noting that basic versions of Bayesian filters were employed.

Finally, we decided to perform additional tests on artificial data. For this purpose, we used the well-known sine and arctangent functions. Then, we added a Gaussian noise parameter to the original data with a standard deviation of 2.0 in part (a) and 0.1 in part (b). Figure 6 shows the application of the APF filter (N = 500) and Kalman filters on synthetic data. The results show the significant advantage of the APF filter over the extended Kalman filter.

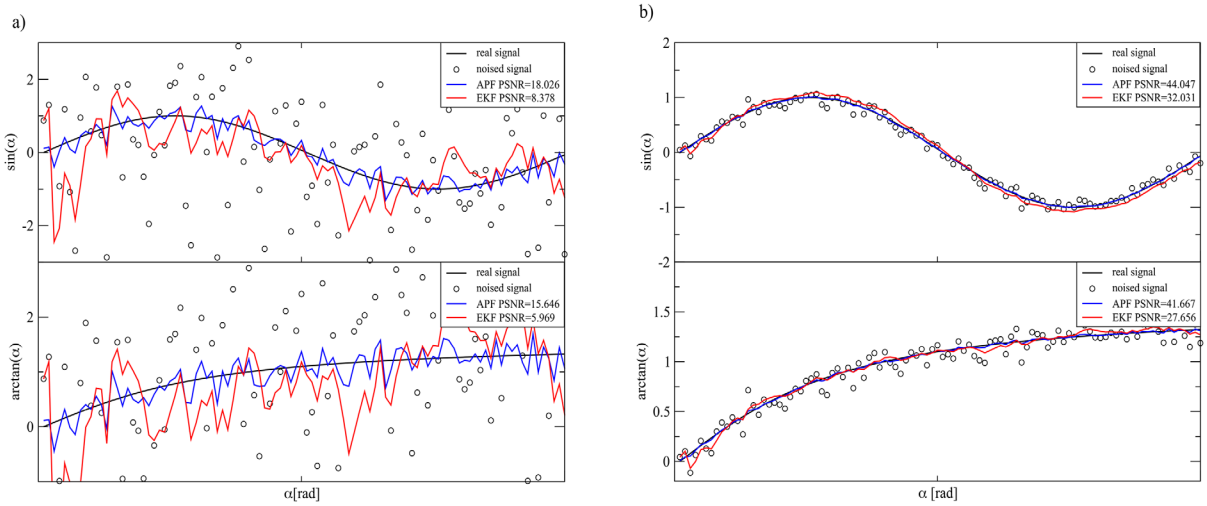

Figure 6. Estimating a random sine and arctangent signal using APF and EKF for a Gaussian noise added to the original data with a standard deviation equal to 2.0 in part (a) and 0.1 in part (b). The value of PSNR coefficients is provided in the legend box.

CONCLUSIONS

Data obtained from an experiment are always burdened with a certain amount of uncertainty. Particle filters are very often used to filter such data. As a result, we obtain data with a reduced level of interference. In a computer experiment, six experimental data sets representing different areas of science were used. For the purpose of noise reduction, four basic Bayesian filters were used: generic particle filter, SIR particle filter, auxiliary particle filter, and regularized particle filter. The peak signal-to-noise ratio measure was used to assess the quality of the level of noise reduction.

The most important results of this study include the following:

- Using Bayesian filters on experimental data significantly reduced the standard deviation values.
- Gradually increasing the number of inserted particles gradually increased the PSNR value until a steady state value was reached.
- The auxiliary particle filter achieved the highest PSNR values. The higher the signal change dynamics, the better the values were achieved.
- All Bayesian filters used showed better ability to denoise nonlinear data than the EKF filter.
- Making the resampling process dependent on the N_T parameter speeds up the calculation time but worsens the quality of the estimation, resulting in lower PSNR values.

REFERENCES

1. Wójcicki P., Zientarski T., Przyłucki S. Application of Particle Filter in Path-loss Modelling. Advances in Science and Technology Research Journal. 2023; 17(5): 337-349. https://doi.org/10.12913/22998624/172407
2. Zhang G. Suh I.H. Integration of a prediction mechanism with a sensor model: An anticipatory Bayes filter. IEEE International Conference on Robotics and Automation. 2009; 3620-3625, Kobe, Japan. https://dl.acm.org/doi/10.5555/1732643.1732695
3. Wang W., Ku W.S. Dynamic indoor navigation with Bayesian filters, SIGSPATIAL Special, 2017; 8(3): 9-10. https://doi.org/10.1145/3100243.3100249
4. Li Q., Qi B., Liang G. Approximate Bayes multitarget tracking smoother. IET Radar, Sonar and Navigation. 2019; 13(3): 428–437. https://doi.org/10.1049/iet-rsn.2018.5297
5. Burghardt D., Lanillos P., Robot Localization and Navigation Through Predictive Processing Using LiDAR. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Cham: Springer International Publishing. 2021; 857-864. https://doi.org/10.1007/978-3-030-93736-2_61
6. Mamot A., Mamot M., Leski J. Bayesian and empirical Bayesian approach to weighted averaging of ECG signal, Bulletin of the Polish Academy of Sciences. Technical Sciences. 2007; 55(4): 341-350.
7. Martin G.M., Frazier D.T., Maneesoonthorn W., Loaiza-Maya R., Huber F., Koop G., Maheu J., Nibbering D., Panagiotelis A. Bayesian forecasting in economics and finance: A modern review, International Journal of Forecasting. 2024; 40(2): 811–839. https://doi.org/10.1016/j.ijforecast.2023.05.002

8. Huang D., Guan G., Zhou J. Network-based naive Bayes model for social network. Science China Mathematics. 2018; 61: 627–40. https://doi.org/10.1007/s11425-017-9209-6.
9. Guo Y., Bai G., Hu Y. Using Bayes Network for Prediction of Type2 Diabetes. In: Proceedings of the 2012 International Conference for Internet Technology and Secured Transactions. London, UK; 2012; 471–72.
10. Daisy SJS., Begum A.R. Smart material to build mail spam filtering technique using Naive Bayes and MRF methodologies. Materials Today: Proceedings 2021; 47: 446–52. https://doi.org/10.1016/j.matpr.2021.04.630
11. Cheng J., Greiner R. Comparing Bayesian network classifiers. arXiv preprint arXiv:1301.6684. 2013. https://doi.org/10.48550/arXiv.1301.6684
12. Sameni R. S., Mohammad B., Jutten C., Clifford, G. D., A Nonlinear Bayesian Filtering Framework for ECG Denoising, IEEE Transactions on Biomedical Engineering. 2007; 54(12): 2172-2185. https://doi.org/10.1109/tbme.2007.897817
13. Lee Y. K., Kwon O. W., Shin H. S., Jo J., Lee Y.. Noise reduction of PPG signals using a particle filter for robust emotion recognition. 2011 IEEE International Conference on Consumer Electronics -Berlin. 2011; 202-205.
14. Frosio I., Olivieri C., Lucchese M., Borghese N. A., Boccacci P. Bayesian denoising in digital radiography: A comparison in the dental field. Computerized Medical Imaging and Graphics. 2006; 37(1): 28-39. http://dx.doi.org/10.1016/j.compmedimag.2012.10.003
15. Kitagawa G. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. J. Comput. Graph. Statist. 1996; 5(1): 1-25. https://doi.org/10.2307/1390750
16. Arulampalam M.S., Maskell S., Gordon N., Clapp T. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Transactions on Signal Processing. 2002; 50(2): 174–188.
17. Torta J., Torta E., Van De Molengraft R. Particle Filters: A Hands-On Tutorial. Sensors. 2021; 21(2): 438. https://doi.org/10.3390/ s21020438
18. Papaioannou I., Papadimitriou C., Straub D. Sequential importance sampling for structural reliability analysis. Structural safety. 2016; 62: 66–75. https://doi.org/10.1016/j.strusafe.2016.06.002
19. Doucet A., Godsill S., Andrieu C. On sequential Monte Carlo methods for Bayesian filtering. Statistics and computing. 2000. 10: 197-208. https://doi.org/10.1023/A:1008935410038
20. Bolić M., Djurić P.M., Hong S. Resampling Algorithms for Particle Filters: A Computational Complexity Perspective. EURASIP Journal on Advances in Signal Processing. 2004; 1-11. https://doi.org/10.1155/S1110865704405149
21. Chopin N., Singh S.S., Soto T., Vihola M. On resampling schemes for particle filters with weakly informative observations. The Annals of Statistic. 2022; 50(6): 3197–222. https://doi.org/10.1214/22-AOS2222
22. Hol J.D., Schon T.B., Gustafsson F. On Resampling Algorithms for Particle Filters. IEEE Nonlinear Statistical Signal Processing Workshop. 2006; 79-82
23. Gordon N., Salmond D., Smith A.F.M. Novel approach to nonlinear and non-Gaussian Bayesian state estimation. Proc. Inst. Elect. Eng. F. 1993; 140: 107–113. https://doi.org/10.1049/ip-f-2.1993.0015
24. Pitt M., Shephard N. Filtering via simulation: Auxiliary particle filters. J. Amer. Statist. Assoc. 1999; 94(446): 590–599.
25. Bruno M.G.S. Regularized Particle Filters. In: Sequential Monte Carlo Methods for Nonlinear Discrete-Time Filtering. Springer International Publishing. 2013; 49-50. https://doi.org/10.1007/978-3-031-02535-8_10
26. Musso, C., Oudjane, N., Le Gland, F. Improving Regularised Particle Filters. In: Doucet, A., de Freitas, N., Gordon, N. (eds) Sequential Monte Carlo Methods in Practice. Statistics for Engineering and Information Science. Springer, New York, NY. 2001. https://doi.org/10.1007/978-1-4757-3437-9
27. Wojcicki P., Zientarski T., Charytanowicz M., Lukasik E. Estimation of the path-loss exponent by Bayesian filtering method. Sensors. 2021; 21(6): 1934. https://doi.org/10.3390/s21061934
28. Zientarski, T., Chocyk D. Stress induced grain boundaries in thin Co layer deposited on Au and Cu. Appl. Phys. A 2016; 122: 908. https://doi.org/10.1007/s00339-016-0432-x
29. Tisov O., Pashechko M., Yurchuk A., Chocyk D., Zubrzycki J., Prus A., Wlazło-Ćwiklińska M. Microstructure and friction response of a novel eutectic alloy based on the Fe-C-Mn-B system. Materials. 2022; 15(24): 9031. https://doi.org/10.3390/ma15249031
30. Yin T.S., Othman A.R., Sulaiman S., Mohamed-Ibrahim M.I., Razha-Rashid M. Application of mean and standard deviation in questionnaire surveys: Construct validation. Jurnal Teknologi. 2016; 78: 99–105. https://doi.org/10.11113/jt.v78.8983
31. Przystupa K., Kolodiy Z., Yatsyshyn S., Majewski J., Khoma Y., Petrovska I., Lasarenko S., Hut T. Standard deviation in the simulation of statistical measurements. Metrology and Measurement Systems. 2023; 30: 17–30.
32. Wang C., Zheng Y., Chang H.H. Does standard deviation matter? Using "standard deviation" to quantify security of multistage testing. Psychometrika.

2014; 79: 154–74. https://doi.org/10.1007/s11336-013-9356-y

33. Macaskill P. Standard deviation and standard error: Interpretation, usage, and reporting. Medical Journal of Australia. 2018; 208(2): 63-64. https://doi.org/10.5694/mja17.00633
34. Barde M.P., Barde P.J. What to use to express the variability of data: Standard deviation or standard error of mean? Perspectives in Clinical Research. 2012; 3(3): 113–16. https://doi.org/10.4103/2229-3485.100662
35. Andrade C. Understanding the Difference Between Standard Deviation and Standard Error of the Mean, and Knowing When to Use Which. Indian Journal of Psychological Medicine. 2020;42(4):409-410. https://doi.org/10.1177/0253717620933419
36. Hodson, T.O. Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not, Geoscientific Model Development. 2022; 15: 5481–5487. https://doi.org/10.5194/gmd-15-5481-2022
37. Clementz B.A., Iacono W.G., Grove W.M. The construct validity of root-mean-square error for quantifying smooth-pursuit eye tracking abnormalities in schizophrenia. Biological Psychiatry. 1996; 39: 448–450. https://doi.org/10.1016/0006-3223(95)00549-8
38. Chai T., Draxler R.R. Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature, Geoscientific Model Development. 2014; 7: 1247–50. https://doi.org/10.5194/gmd-7-1247-2014
39. Huynh-Thu Q., Ghanbari M. The accuracy of PSNR in predicting video quality for different video scenes and frame rates, Telecommunication Systems. 2012; 49: 35-48 https://doi.org/10.1007/s11235-010-9351-x
40. Huynh-Thu Q., Ghanbari M. Scope of validity of PSNR in image/video quality assessment, Electronics Letters. 2008; 44: 800–801. https://doi.org/10.1049/el:20080522
41. Lian J. Image Sharpening with Optimized PSNR. Proceedings of the 2019 IEEE International Conference on Signal and Image Processing Applications(ICSIPA). 2019; 106–10.
42. Turaga D.S., Chen Y.W., Caviedes J. No reference PSNR estimation for compressed pictures. Signal Processing-Image Communication 2004; 19: 173–84. https://doi.org/10.1016/j.image.2003.09.001
43. Keles O., Yilmaz M.A., Tekalp A.M., Korkmaz C., Dogan Z. On the computation of PSNR for a set of images or video. Proc. 2021 Picture Coding Symposium (PCS). 2021; 286–90. https://doi.org/10.48550/arXiv.2104.14868
44. Szymczyk T., Czajka P. Analysis of the possibility of hiding decomposed information in the virtual reality environment. Advances in Science and Technology Research Journal. 2025; 19(2):283-295. https://doi.org/10.12913/22998624/195718
45. Kozieł G., Malomuzh L. 3D Model Fragile Watermarking Scheme for Authenticity Verification. Advances in Science and Technology Research Journal. 2024; 18(8):351-365. https://doi.org/10.12913/22998624/194146
46. Gentle, J.E. Random number generation and Monte Carlo methods. Vol. 381. New York: Springer, 2003. https://doi.org/10.1007/978-1-4757-2960-3
47. Vetterling, W.T. Numerical recipes example book (C) (2nd ed., repr). Cambridge University Press, 1997
48. Im, G. Notes on Kalman Filter (KF, EKF, ESKF, IEKF, IESKF). arXiv preprint arXiv:2406.06427, 2024. https://doi.org/10.48550/arXiv.2406.06427